Operational Research
https://doi.org/10.1007/s12351-022-00735-9

ORIGINAL PAPER

The two-dimensional cutting stock problem with usable leftovers: mathematical modelling and heuristic approaches

Douglas Nogueira do Nascimento[1] · **Adriana Cristina Cherri**[1] · **José Fernando Oliveira**[2]

Received: 14 July 2021 / Revised: 24 June 2022 / Accepted: 22 July 2022

Abstract
Different variations of the classic cutting stock problem (CSP) have emerged and presented increasingly complex challenges for scientists and researchers. One of these variations, which is the central subject of this work, is the two-dimensional cutting stock problem with usable leftovers (2D-CSPUL). In these problems, leftovers can be generated to reduce waste. This technique has great practical importance for many companies, with a strong economic and environmental impact. In this paper, a non-linear mathematical model and its linearization are proposed to represent the 2D-CSPUL. Due to the complexity of the model, a heuristic procedure was also proposed. Computational tests were performed with instances from the literature and randomly generated instances. The results demonstrate that the proposed model and the heuristic procedure satisfactorily solve the problem, proving to be adequate and beneficial tools when applied to real situations.

Keywords Two-dimensional cutting stock problem · Usable leftovers · Mathematical modelling · Exact methods · Heuristic procedure

✉ Douglas Nogueira do Nascimento
douglas.nogueira@unesp.br

Adriana Cristina Cherri
adriana.cherri@unesp.br

José Fernando Oliveira
jfo@fe.up.pt

[1] São Paulo State University (UNESP), Bauru, Brazil

[2] University of Porto, Porto, Portugal

Published online: 06 August 2022

1 Introduction

The cutting stock problem (CSP) is a well-known problem in the literature, with practical applications in industries where minimizing the waste of raw materials is critical for their economic and environmental performance. Considering two dimensions, the problem consists of cutting a set of standard rectangular plates available in stock to produce smaller rectangular items. The cutting process must be planned to meet a known demand while minimizing an objective function, such as material waste or the number of plates used, among others. One of the first studies to address the classical two-dimensional CSP was Gilmore and Gomory (1965), which looked at an extension of the column generation technique proposed by Gilmore and Gomory (1961, 1963) for the one-dimensional CSP.

In addition to the inherent difficulty of solving cutting problems, two-dimensional problems have the geometric complexity of ensuring that the items totally fit inside the plates and do not overlap. That becomes a major challenge for companies regarding production planning, and it increases the demand for scientific research to assist in this process.

An effective strategy usually used to reduce waste in the cutting problems consists of generating usable leftovers during the cutting process. Usable leftovers are pieces with predefined dimensions that are returned to stock to be cut into items in the future. Therefore, these leftovers are not considered waste. The actual dimensions of leftovers are defined on a case-by-case basis based on the order history of the company. Before being generated, usable leftovers must be planned to have a high probability of use in future cutting processes, providing several advantages for companies, as discussed in Coelho et al. (2017). For two-dimensional problems, this variation is known as the two-dimensional cutting stock problem with usable leftovers (2D-CSPUL). An illustrative example of this problem is presented in Sect. 3 (Figs. 1, 2, 3).

Considering the possibility of usable leftovers, most papers addressed the one-dimensional problem, as can be seen in Cherri et al. (2014), where a survey of the existing papers on the CSPUL are presented. After this survey, Arenales et al. (2015), Liu et al. (2017), Tomat and Gradisar (2017) and do Nascimento et al. (2020) also proposed methods to solve the one-dimensional CSPUL.

Due to the importance and applicability of the two-dimensional problems involving usable leftovers, some papers have been published with different approaches to the problem, as can be seen in Cherri (2009), Andrade et al. (2014), Andrade et al. (2016), Clautiaux et al. (2019), Birgin et al. (2020) and Li et al. (2022).

The present paper contributes to the literature by proposing a mathematical model to represent the 2D-CSPUL. The first version of the model has non-linear constraints that are tackled by a linearization strategy. The model uses the strip concept to build guillotine cutting patterns. Guillotine cuts are applied orthogonally from one edge to the other of the plate, dividing it into two parts. Furthermore, the model considers the possibility of two or more items being combined to create new items. Cutting patterns are limited to generating at most one usable leftover obtained from a horizontal guillotined cut. The proposed model solutions were compared with two mathematical models proposed in the literature. All these models were solved through an exact solver.

Another contribution of this paper is the proposal of a heuristic procedure that decomposes the original problem into smaller problems, reducing the computational time to obtain satisfactory integer solutions for large instances. For all the approaches, tests were performed with instances from the literature and randomly generated.

The remainder of this paper is organized as described here. Section 2 presents a relevant review of the literature. In Sect. 3, the 2D-CSPUL is defined and the proposed mathematical model is presented. The strategy used to linearize the model is given in Sect. 4. The heuristic procedure is described in Sect. 5. In Sect. 6, the results of the computational tests performed are shown, and conclusions are presented in Sect. 7.

2 Literature review

The literature for the 2D-CSP is extensive and has presented exact and non-exact solution methods to solve the problem. An exact solution method was proposed by Christofides and Hadjiconstantinou (1995) that presented an exact tree search algorithm considering a maximum number of times a type of item could appear in each cutting pattern. Silva et al. (2010) proposed an integer programming model in which the decision variables indicated whether a particular item was cut from a plate or from a usable leftover. The proposed model considered 2- and 3-stages, exact and non-exact problems with item rotation and it was solved by a commercial solver.

Furini and Malaguti (2013) proposed three mixed integer programming models, two of which had a polynomial and pseudo-polynomial number of variables, allowing them to be solved by a commercial solver. The third model had an exponential number of variables, being solved through branch-and-price techniques. In Furini et al. (2016) a framework for modelling guillotine constraints in two-dimensional cutting problems was given. The authors focused on the Knapsack Problem, proposing a mathematical model and an exact procedure to solve it. Extensions for the 2D-CSP and the Strip Packing Problem were also presented. Kwon et al. (2019) analyzed integer programs based on pattern-based models for the 2D-CSP considering 2-stage guillotine problems. First, a theoretical analysis of the strength of lower bounds obtained from the LP-relaxations of full-pattern and staged-pattern models was presented. The authors also performed computational experiments to analyze the trade-off between the required computation time to solve the LP-relaxations and the strength of the lower bounds. Martin et al. (2020) addressed the 2D-CSP considering plates with defects and a maximum number of times that each type of item could appear in each cutting pattern. The authors proposed an integer linear model that was solved using algorithms based on Benders decomposition and constraint programming.

For non-exact methods, Wang (1983) proposed two combinatorial methods that solve the 2D-CSP by generating constrained cutting patterns from successive horizontal and vertical allocations of items, using a parameter that limits the maximum acceptable percentage of waste in each cutting pattern. Suliman (2006) proposed a three-step heuristic procedure to solve the 2D-CSP. In the first step, the authors solve a one-dimensional problem to define a set of width-cutting patterns with minimum trim loss. In the second stage, for each width-cutting pattern, the same strategy is used to determine the length of the two-dimensional cutting pattern. And in the third step, the frequency of each cutting pattern is defined.

Cintra et al. (2008) proposed algorithms based on dynamic programming and a column generation technique to solve the 2D-CSP, the rectangular Knapsack Problem and the Strip Packing Problem. The proposed strategies allowed the orthogonal rotation of items and the generation of 2-, 3-, and 4-staged cutting patterns. Furini et al. (2012) aimed to improve the solutions found by Cintra et al. (2008) by presenting a heuristic algorithm also based on the column generation technique, whose subproblem is the solution of a two-dimensional Knapsack Problem that only allows 2-staged guillotine cuts.

Dusberger and Raidl (2015) used the Variable Neighborhood Search (VNS) heuristic to find solutions for the 2D-CSP considering 3-staged cuts. The proposed strategy consisted of destroying part of the incumbent solutions and then rebuilding them through dynamic programming. Bouaine et al. (2018) applied the 2D-CSP to a real case study in the furniture industry. The authors proposed a two-step heuristic procedure that consists in (i) generating feasible cutting patterns through classic heuristics of the Bin-Packing Problem, and (ii) solving an integer linear program with the generated cutting patterns to minimize the waste of wood. Wuttke and Heese (2018) proposed a sequential heuristic with a feedback loop to solve the 2D-CSP with sequence-dependent setup times. The heuristic was based on the approach presented by Gilmore and Gomory (1961; 1965), and the authors verified its performance through computational experiments using data from a textile company. Wang et al. (2020) presented an integer programming formulation for the 2D-CSP considering setup costs and the skiving process and solved its linear programming relaxation through a column-and-row generation framework. The authors also proposed a diving heuristic to obtain integer solutions.

Using the strategy of generating leftovers, Cherri (2009) modified the AND/OR graph approach proposed by Morabito (1989) and other heuristic procedures from the literature to solve the problem. Although the waste of material decreased with the possibility of generating leftovers, too many leftovers were generated in each cutting pattern. Andrade et al. (2014) proposed multilevel mathematical programming models to represent the non-guillotine 2D-CSPUL. The authors reformulate these multilevel models as one-level mixed integer programming models and solve them using a commercial exact solver.

Andrade et al. (2016) presented two bilevel mathematical models to solve the non-exact 2-staged guillotine 2D-CSPUL. The first model considers the problem as a bin-packing problem and the second model groups items of the same type. These bilevel models were reformulated as two one-level models that find solutions minimizing the cost of cut objects and, among these solutions with minimum cost, choose the one that maximizes the value of generated leftovers.

Clautiaux et al. (2019) studied the 2D-CSPUL in the glass industry, considering consecutive production batches and aiming to minimize the total width of the cut plates. The authors proposed a diving heuristic based on a column generation technique to generate guillotine cutting patterns. Dynamic programming is used to solve the pricing problem, and leftovers are only allowed in the last cutting pattern for a batch.

Birgin et al. (2020) extended the formulation proposed by Andrade et al. (2014) to consider the multiperiod problem. The extended multiperiod framework for the non-guillotine 2D-CSPUL generates leftovers by horizontal and vertical guillotine precuts, and the objective function is to minimize the cost of the cut objects. Sumetthapiwat et al. (2020) proposed an approach to the 2D-CSPUL that considers multiple stock sizes and leftovers with predefined fixed values. The authors used a column generation technique to find a solution that minimizes the total waste.

Li et al. (2022) studied the multiperiod 2D-CSPUL applied to a company that produces transformers by cutting silicon steel coils. A sequential leftovers utilization correction (SLUC) algorithm was proposed to solve the problem considering several operational constraints of the company, such as the maximum allowed number of leftovers in stock, the minimum and maximum length for cutting patterns and the limited number of cutting knives.

Table 1 provides a summary of the publications described in this section. The columns of this table indicate the variants of the 2D-CSP studied in each paper, regarding the consideration of usable leftovers ("Left"), the solution method (exact or heuristic), the types of cuts (guillotine or non-guillotine), and the number of cutting stages (2-, 3-, 4-, *k*- or unrestricted).

The present paper deals with the guillotine 3-stage 2D-CSPUL. This problem was solved by both an exact and heuristic approach. Among the studies mentioned in this section, we compare the solutions obtained by the proposed approaches with those from the mathematical models proposed by Andrade et al. (2016) and Furini et al. (2016). The aim of comparing with the models from Andrade et al. (2016), which considered the 2-stage 2D-CSPUL, is to analyze the advantages obtained when allowing 3-stage cutting patterns. Regarding the model proposed by Furini et al. (2016), despite not using leftovers, it was chosen for comparison because it is well known in the literature for its excellent performance. For a fair comparison, adaptations were necessary since Furini's model does not restrict the number of cutting stages.

3 Problem definition and mathematical formulation

In the 2D-CSPUL, a set of standard rectangular plates must be cut to produce a set of smaller rectangular items minimizing an objective function. This cutting process must consider the possibility of planning the generation of leftovers that are not considered waste and are kept in stock to be used in future cutting processes.

Figure 1 illustrates the 2D-CSPUL. In this example, there is one type of standard plate, with width $W = 55$, height $H = 38$ and availability in stock $e = 2$. There is demand for 5 types of items with dimensions (width × height): Item $1 = 8 \times 15$, Item

Table 1 Summary of papers in the literature review

References		Solution method		Cuts		Stages				
	Left.	Exact	Heur.	Guill.	Non-guill.	2-	3-	4-	k-	Unrest.
Wang (1983)			✓	✓						✓
Christofides and Hadjiconstantinou (1995)		✓		✓					✓	
Suliman (2006)			✓	✓						✓
Cintra et al. (2008)			✓	✓					✓	
Cherri (2009)	✓		✓	✓		✓				
Silva et al. (2010)		✓		✓		✓	✓			
Furini et al. (2012)			✓	✓		✓				
Furini and Malaguti (2013)		✓		✓		✓				
Andrade et al. (2014)	✓	✓			✓					✓
Dusberger and Raidl (2015)			✓	✓			✓			
Andrade et al. (2016)	✓	✓		✓		✓				
Furini et al. (2016)		✓		✓						✓
Bouaine et al. (2018)			✓		✓					✓
Wuttke and Heese (2018)			✓	✓						✓
Clautiaux et al. (2019)	✓		✓	✓				✓		
Kwon et al. (2019)		✓		✓		✓				
Birgin et al. (2020)	✓	✓			✓					✓
Martin et al. (2020)		✓		✓						✓
Sumetthapiwat et al. (2020)	✓	✓	✓	✓		✓				
Wang et al. (2020)			✓	✓		✓				
Li et al. (2022)	✓		✓	✓		✓				
Present paper	✓	✓	✓	✓			✓			

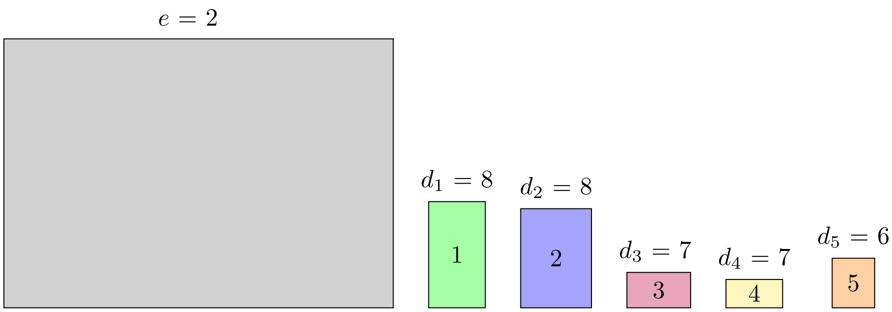


Fig. 1 Example of a 2D-CSPUL

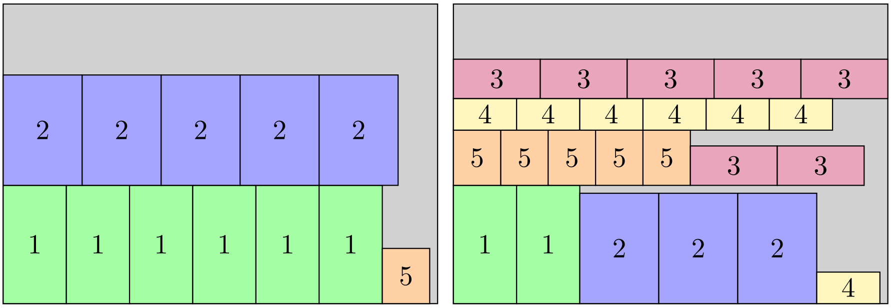


Fig. 2 Possible solution for the 2D-CSPUL without a usable leftover

2 = 10 × 14, Item 3 = 9 × 5, Item 4 = 8 × 4 and Item 5 = 6 × 7. These items must be produced in the quantities $d_1 = 8$, $d_2 = 8$, $d_3 = 7$, $d_4 = 7$ and $d_5 = 6$.

Figure 2 shows a possible solution with two cutting patterns that meet the demand and generate waste of 31.32%.

By considering the possibility of generating one usable leftover, items can be reallocated to concentrate a larger number of them in one cutting pattern. This strategy increases the available space in the other cutting pattern and allows the production of a leftover with dimensions of 55 × 16. Figure 3 shows the new solution, which reduces the waste to 10.26%.

Cutting patterns in Figs. 2 and 3 were built using the strip concept (Lodi and Monaci 2003) and are 2-stage cutting patterns. In the next section, the strategy used to build cutting patterns and the proposed mathematical model are presented.

3.1 Mathematical model

The mathematical programming model formulated for the 2D-CSPUL uses the indices and parameters presented in Table 2. We consider that different types of standard plates and leftovers are available in stock. Standard plates can produce both items and leftovers. But leftovers in stock can produce only items. For convenience, the types of items are sorted by descending order of height ($h_1 \geq h_2 \geq h_3 \geq \cdots \geq h_I$).

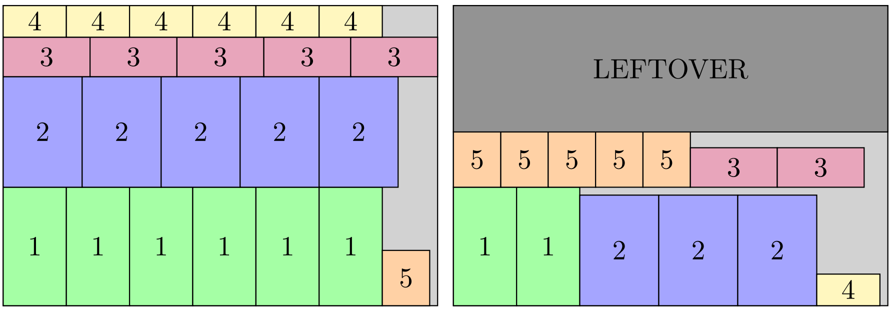


Fig. 3 Possible solution for the 2D-CSPUL generating a usable leftover

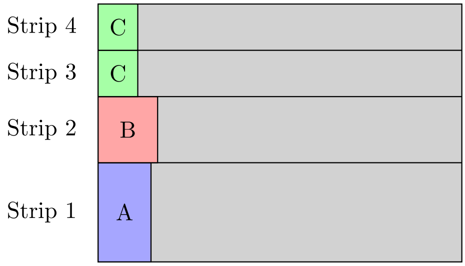
Table 2 List of indices and parameters used by the proposed mathematical model

	Description
Index	
$s = 1, \dots, S$	Number of types of standard plates in stock
$r = 1, \dots, R$	Number of types of leftovers in stock
$v = 1, \dots, S + R$	Number of types of plates (standard and leftovers) in stock
$i = 1, \dots, I$	Number of types of ordered items
$k = 1, \dots, K$	Number of types of macro-items
$f = 1, \dots, F_v$	Number of strips for plate v
$\eta = 1, \dots, H^F$	Number of heights for strips
$j = 1, \dots, J_v$	Maximum number of cutting patterns for plate v
Parameter	
W_v	Width of a plate of type v
H_v	Height of a plate of type v
e_v	Number of plates of type v available in stock
w_i	Width of item i
h_i	Height of item i
d_i	Demand for item i
A_{iv}	Maximum number of items i allocated to a strip in a plate of type v
A^N_{kv}	Maximum number of macro-items k allocated to a strip in a plate of type v
w^K_k	Width of macro-item k
h^K_k	Height of macro-item k
n_{ik}	Number of items i in macro-item k
h^F_η	Height η for strips
$m_{i\eta}$	Binary parameter that indicates if the height of item i is equal to or lower than height η
$m^K_{k\eta}$	Binary parameter that indicates if the height of macro-item k is equal to or lower than height η
h^R_{min}	Minimum height for generated leftovers
h^R_{max}	Maximum height for generated leftovers
U	Maximum number of generated leftovers
α	Reuse rate of generated leftovers

The proposed model builds cutting patterns using the strip concept. For each cutting pattern j, the plate is divided into horizontal strips whose heights are equal to the height of the highest item in each strip. If an item i is the highest item in strip f, then item i initializes strip f. Each type of item can initialize one or more strips in a cutting pattern.

After each strip is initialized, the remaining items whose heights are equal to or lower than the height of the strip can be allocated on the right side of the last item already allocated. Rotation of an item is not considered. Figure 4 illustrates an example of a cutting pattern that divides the plate into four strips.

Fig. 4 Standard plates divided into strips



A strategy proposed for the creation of cutting patterns is the possibility of generating *macro-items*. A macro-item is composed of two or more items allocated one above the other, since the resulting height does not exceed the height of the largest ordered item. Each possible combination of items following this constraint is considered a new type of item. This strategy of combining items was first addressed by Wang (1983), which allowed the vertical and horizontal combination between each pair of items, as long as the percentage of waste generated in the constructed rectangle formed by the items does not exceed a maximum value. This condition is also used in this paper.

Figure 5 shows a constructed rectangle formed by two items with dimensions (11×5) and (6×7). The waste generated in this constructed rectangle is 26.51% of the area.

Using macro-items considerably increases the problem complexity and the computational resolution time since it becomes a 3-staged cutting problem. However, macro-items allow the minimization of material loss by enhancing the diversity of items, making it possible to create better-cutting patterns and find better solutions. This trade-off between solving time and reduced waste is analyzed in Sect. 6, dedicated to the computational experiments.

For a set of ordered items, K types of macro-items are defined. The height h_k^K of a macro-item k is the sum of the heights of all items in k. Considering n_{ik} the number of items i in macro-item k, then h_k^K is calculated as follows:

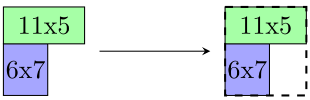
$$h_k^K = \sum_{i=1}^{I} n_{ik} h_i.$$

The width w_k^K of each macro-item k is equal to the maximum width between all items in k, and is defined as follows:

$$w_k^K = \max \left\{ w_i \mid n_{ik} > 0 \right\}.$$

Figure 6 illustrates a situation with $I = 4$ types of items and $K = 5$ possible types of macro-items. The height of the largest item is $h_1 = 15$, and this is the upper bound

Fig. 5 Example of constructed rectangle for a macro-item



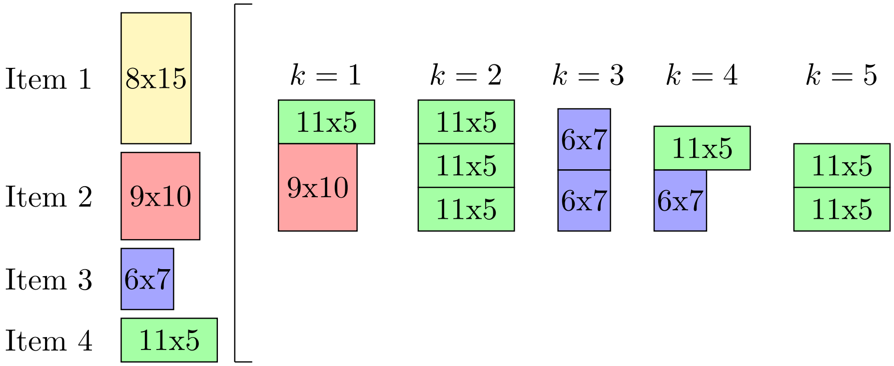


Fig. 6 Example of macro-items for an instance

for the height of macro-items. This implies that the item $i = 1$ will never be in a macro-item.

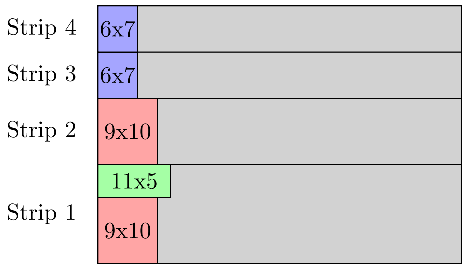
In Fig. 6, the parameters regarding the macro-items have the values:

- Number of types of macro-items: $K = 5$.
- Heights of macro-items: $h_k^K = [15\ 15\ 14\ 12\ 10]$.
- Widths of macro-items: $w_k^K = [11\ 11\ 6\ 11\ 11]$.
- Number of each type of item in each macro-item: $n_{ik} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 1 & 0 \\ 1 & 3 & 0 & 1 & 2 \end{bmatrix}$.

Since macro-items are new types of items, they can also initialize strips in cutting patterns. Therefore, there is a set of H^F possible heights for strips, including the heights of items and macro-items. In Fig. 6, $H^F = 6$ can be represented by the vector $h_\eta^F = (15, 14, 12, 10, 7, 5)$. Figure 7 uses the same items as Fig. 6 and illustrates a cutting pattern with one strip initialized by a macro-item and three strips initialized by items.

After a height η is assigned to a strip f, items and macro-items can be allocated to f as long as their heights are not greater than h_η^F. In order to keep the linearity of this constraint, parameters $m_{i\eta}$ and $m_{k\eta}^K$ are used to indicate, respectively, whether an item i or a macro item k can be allocated in a strip with height η. These parameters are defined as 1 when the height of item i or macro-item k is equal to or lower than height η and are defined as 0 otherwise.

Fig. 7 Different items and macro-items initializing strips in a cutting pattern



Regarding leftovers, each cutting pattern j can generate, at most, one leftover from a standard plate s. The leftover must have the same width as the standard plate, and height varying within the interval $[h^R_{min}H_s, h^R_{max}H_s]$, where h^R_{max} and h^R_{min} are values between 0 and 1, and $h^R_{max} > h^R_{min}$.

Although the leftovers are generated to reduce the waste at the current moment in time, it is not possible to guarantee that they will be used entirely in future cutting processes. Thus, the parameter $0 \leq \alpha \leq 1$ indicates the reuse rate of leftovers, which limits the impact of the generation of leftovers in the objective function.

Due to the characteristics of the problem, some constraints in the proposed model are non-linear. Next, the complete mathematical model is presented.

3.1.1 Variables

- $y_{\eta fjv} = \begin{cases} 1, & \text{if height } \eta \text{ is assigned to strip } f \text{ in cutting pattern } j \text{ for a} \\ & \text{plate of type } v. \\ 0, & \text{otherwise.} \end{cases}$
- a_{ifjv}: number of items i allocated to strip f in cutting pattern j for a plate of type v;
- a^K_{kfjv}: number of macro-items k allocated to strip f in cutting pattern j for a plate of type v;
- x_{jv}: number of plates of type v cut according to cutting pattern j;
- $g_{js} = \begin{cases} 1, & \text{if a leftover is generated in cutting pattern j for a standard} \\ & \text{plate of type s.} \\ 0, & \text{otherwise.} \end{cases}$
- l_{js}: height of leftover generated in cutting pattern j for a standard plate of type s.

3.1.2 Mathematical model

$$\text{Min} \ : \sum_{v=1}^{S+R} \sum_{j=1}^{J_v} W_v H_v x_{jv} - \alpha \sum_{s=1}^{S} \sum_{j=1}^{J_s} W_s l_{js} x_{js} \tag{1}$$

$$\text{Subject to} \ : \sum_{\eta=1}^{H^F} y_{\eta fjv} \leq 1, \qquad \forall f, j, v \tag{2}$$

$$\sum_{\eta=1}^{H^F} \sum_{f=1}^{F_s} h^F_\eta y_{\eta fjs} + l_{js} \leq H_s, \qquad \forall j, s \tag{3}$$

$$\sum_{\eta=1}^{H^F} \sum_{f=1}^{F_r} h^F_\eta y_{\eta fjr} \leq H_r, \qquad \forall j, r \tag{4}$$

$$\sum_{i=1}^{I} w_i a_{ifjv} + \sum_{k=1}^{K} w_k^K a_{kfjv}^K \le W_v, \quad \forall f, j, v \tag{5}$$

$$a_{ifjv} \le A_{iv} \sum_{\eta=1}^{H^F} m_{i\eta} y_{\eta fjv}, \qquad \forall i, f, j, v \tag{6}$$

$$a_{kfjv}^K \le A_{kv}^N \sum_{\eta=1}^{H^F} m_{k\eta}^K y_{\eta fjv}, \qquad \forall k, f, j, v \tag{7}$$

$$\sum_{v=1}^{S+R} \sum_{f=1}^{F_v} \sum_{j=1}^{J_v} a_{ifjv} x_{jv} + \sum_{k=1}^{K} \sum_{v=1}^{S+R} \sum_{f=1}^{F_v} \sum_{j=1}^{J_v} n_{ik} a_{kfjv}^K x_{jv} = d_i, \quad \forall i \tag{8}$$

$$\sum_{j=1}^{J_v} x_{jv} \le e_v, \qquad \forall v \tag{9}$$

$$\sum_{s=1}^{S} \sum_{j=1}^{J_s} g_{js} x_{js} \le U \tag{10}$$

$$h_{min} H_s g_{js} \le l_{js} \le h_{max} H_s g_{js} \qquad \forall j, s \tag{11}$$

$$y_{\eta fjv} \in \{0, 1\}, \qquad \forall \eta, f, j, v \tag{12}$$

$$a_{ifjv} \in [0, A_{iv}] \text{ and integer}, \qquad \forall i, f, j, v \tag{13}$$

$$a_{kfjv}^K \in \left[0, A_{kv}^N\right] \text{ and integer}, \qquad \forall k, f, j, v \tag{14}$$

$$x_{jv} \in [0, e_v] \text{ and integer}, \qquad \forall j, v \tag{15}$$

$$g_{js} \in \{0, 1\}, \qquad \forall j, s \tag{16}$$

$$l_{js} \in Z, \qquad \forall j, s \tag{17}$$

In the model (1)–(17), the objective function (1) minimizes the total area of cut plates, excluding the area of the generated leftovers. Constraint (2) ensures that only one height is assigned to each initialized strip. Constraints (3) and (4) guarantee that the sum of the heights of all initialized strips and generated leftovers is viable for the plate. Constraint (5) ensures that the sum of the widths of items and macro-items allocated to each strip is equal to or lower than the width of the cut object.

Constraints (6) and (7) prevent items and macro-items respectively from being allocated to strips with heights lower than their own heights. Constraint (8) ensures that the demand is satisfied and the stock constraint is given by (9). Constraint (10) limits the quantity of generated leftovers. Constraint (11) determines the minimum and maximum height of generated leftovers. Constraints (12)–(17) are the integrality and non-negativity constraints of the variables.

Due to the objective function (1) and the constraints (8) and (10), the proposed model is non-linear. Therefore, a linearization strategy was applied to the model. The details of this strategy are described in Sect. 4.

4 Linearization strategy

To describe the linearization strategy applied to the model (1)–(17), two decision variables are necessary. These variables are integer p and binary q, such that $0 \leq p \leq M$, with M being a sufficiently large value. The multiplication pq, that results in non-linearity, can be replaced by only one positive integer variable z if the following linear constraints are add to the problem:

$$\begin{aligned} &Mq \geq z \\ &p \geq z \\ &p - M(1-q) \leq z. \end{aligned}$$

In the proposed model, the objective function (1) and the demand constraint (8), include the product between integer variables ($l_{js}x_{js}$, $a_{ifjv}x_{jv}$ and $a^K_{kfjv}x_{jv}$). To linearize them, the integer variable x_{jv} was converted to binary. The new definition of variable x_{jv}, $\beta = 1, \dots, e_v$, is:

$$x_{\beta jv} = \begin{cases} 1, \text{if } \beta \text{ plates of type } v \text{ are cut according to cutting pattern } j. \\ 0, \text{otherwise.} \end{cases}$$

With this modification, the multiplication $l_{js}x_{\beta js}$ can be replaced by an integer variable $z^O_{\beta js}$. The new linear objective function is:

$$\text{Min} \ : \sum_{v=1}^{S+R} \sum_{j=1}^{J_v} \sum_{\beta=1}^{e_v} \beta W_v H_v x_{\beta jv} - \alpha \sum_{s=1}^{S} \sum_{j=1}^{J_s} \sum_{\beta=1}^{e_s} \beta W_s z^O_{\beta js}. \tag{18}$$

To guarantee that this linear objective function (18) is equivalent to the non-linear objective function (1), the following constraints are added to the model:

$$h_{max} H_s x_{\beta js} \geq z^O_{\beta js}, \qquad \forall \beta, j, s \tag{19}$$

$$l_{js} \geq z^{O}_{\beta js}, \qquad \forall \beta, j, s \tag{20}$$

$$l_{js} - h_{max} H_s (1 - x_{\beta js}) \leq z^{O}_{\beta js}, \qquad \forall \beta, j, s. \tag{21}$$

By using the same strategy, the multiplications $a_{ifjv} x_{\beta jv}$ and $a^{K}_{kfjv} x_{\beta jv}$ in the non-linear demand constraint (8) can be replaced by integer variables $z^{I}_{\beta ifjv}$ and $z^{K}_{\beta kfjv}$, respectively. The new linear demand constraint is:

$$\sum_{v=1}^{S+R} \sum_{f=1}^{F_v} \sum_{j=1}^{J_v} \sum_{\beta=1}^{e_v} \beta z^{I}_{\beta ifjv} + \sum_{k=1}^{K} \sum_{v=1}^{S+R} \sum_{f=1}^{F_v} \sum_{j=1}^{J_v} \sum_{\beta=1}^{e_v} \beta n_{ik} z^{K}_{\beta kfjv} = d_i, \quad \forall i. \tag{22}$$

The equivalence between the linear demand constraint (22) and non-linear demand constraint (8) is guaranteed by adding the following constraints to the model:

$$A_{iv} x_{\beta jv} \geq z^{I}_{\beta ifjv}, \qquad \forall \beta, i, f, j, v \tag{23}$$

$$a_{ifjv} \geq z^{I}_{\beta ifjv}, \qquad \forall \beta, i, f, j, v \tag{24}$$

$$a_{ifjv} - A_{iv}(1 - x_{\beta jv}) \leq z^{I}_{\beta ifjv}, \qquad \forall \beta, i, f, j, v \tag{25}$$

$$A^{N}_{kv} x_{\beta jv} \geq z^{K}_{\beta kfjv}, \qquad \forall \beta, k, f, j, v \tag{26}$$

$$a^{K}_{kfjv} \geq z^{K}_{\beta kfjv}, \qquad \forall \beta, k, f, j, v \tag{27}$$

$$a^{K}_{kfjv} - A^{N}_{kv}(1 - x_{\beta jv}) \leq z^{K}_{\beta kfjv}, \qquad \forall \beta, k, f, j, v. \tag{28}$$

For the linearization of constraint (10), the original multiplication between a binary and an integer variable ($g_{js} x_{js}$) needs to be removed. With the new definition of the frequency variable, ($x_{\beta js}$), there is the multiplication of two binary variables. This multiplication can be replaced by a single binary variable $b_{\beta js}$, adding the following linear constraints to the model:

$$g_{js} \geq b_{\beta js}, \qquad \forall \beta, j, s \tag{29}$$

$$x_{\beta js} \geq b_{\beta js}, \qquad \forall \beta, j, s \tag{30}$$

$$g_{js} + x_{\beta js} - 1 \leq b_{\beta js}, \qquad \forall \beta, j, s. \tag{31}$$

Constraints (29) and (30) together ensure that if at least one of the binary variables is 0, then $b_{\beta js}$ will also be 0. In the case of both variables being 1, the left-hand side of Constraint (31) will be 1, forcing $b_{\beta js}$ to also be 1. Thus, non-linear Constraint (10) can be replaced by the following linear constraint:

$$\sum_{s=1}^{S}\sum_{j=1}^{J_s}\sum_{\beta=1}^{e_s}\beta b_{\beta js} \leq U. \tag{32}$$

Even though the stock constraint (9) of the non-linear model is linear, it needs to be changed due the definition of the frequency variable $x_{\beta js}$. The stock constraint of the linear model is:

$$\sum_{j=1}^{J_v}\sum_{\beta=1}^{e_v}\beta x_{\beta jv} \leq e_v, \qquad \forall v. \tag{33}$$

The proposed linear mathematical model can be solved by commercial solvers. However, due to the exponential number of decision variables and integrality constraints, it can be impracticable for these solvers to find optimal solutions for some large instances. Besides this mathematical model, a heuristic procedure to find good quality solutions in a reasonable computational time was developed. This heuristic procedure is described in Sect. 5.

5 Heuristic procedure

As defined, the parameter J_v represents the maximum number of different cutting patterns that can be generated for each type of plate v. This value can be, at most, e_v. Depending on the number of plates in stock, it is costly for the model to generate too many cutting patterns and determine their frequencies simultaneously. Since most decision variables of the model refer to the construction of cutting patterns, a two-step heuristic is proposed. This procedure initially generates a set of cutting patterns for all types of plates in stock (Step 1) and then solves an adaptation of the model using these cutting patterns, allowing the creation of just one cutting pattern at each iteration (Step 2) to complete the demand. These two steps are detailed next, and Fig. 8 presents a flowchart that describes the proposed heuristic procedure.

5.1 Step 1: creating initial cutting patterns

In the first step of the heuristic, a set of initial cutting patterns C_v is created for each plate type v in stock. These cutting patterns are created based on a greedy strategy that finds the best cutting patterns for each type of plate in terms of waste, by solving a sub-problem *ST1* that is an adaptation of the proposed mathematical model.

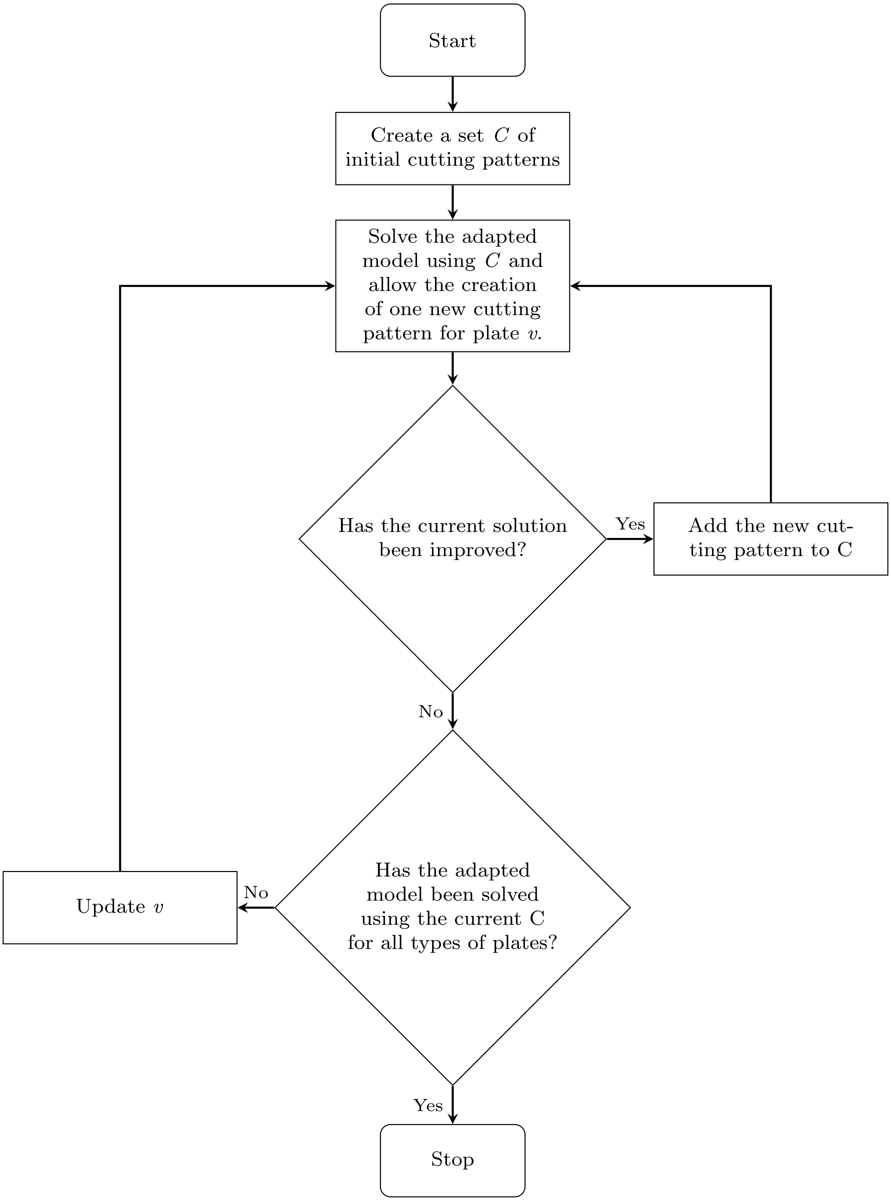


Fig. 8 Flowchart of the proposed heuristic procedure

The sub-problem *ST1* is solved several times, and each iteration generates only one new cutting pattern. Therefore, the decision variable $x_{jv} = 1$.

To guarantee that only one cutting pattern is created, the parameters J and e are limited to $\sum_{v=1}^{S+R} J_v = 1$ and $\sum_{v=1}^{S+R} e_v = 1$ in each iteration of the sub-problem. To indicate if a leftover must be included in a cutting pattern, the parameter U is set to 0 or 1 and the constraint (10) is replaced by:

$$\sum_{s=1}^{S} \sum_{j=1}^{J_s} g_{js} = U. \tag{34}$$

The objective function of the sub-problem *ST1* minimizes the waste generated and is defined as follows:

$$\begin{aligned} \text{Min } : & \sum_{v=1}^{S+R} \sum_{j=1}^{J_v} W_v H_v - \sum_{s=1}^{S} \sum_{j=1}^{J_s} W_s l_{js} \\ & - \sum_{v=1}^{S+R} \sum_{j=1}^{J_v} \left(\sum_{i=1}^{I} \sum_{f=1}^{F_v} w_i h_i a_{ifjv} + \sum_{k=1}^{K} \sum_{i=1}^{I} \sum_{f=1}^{F_v} w_i h_i n_{ik} a_{kfjv}^{K} \right). \end{aligned} \tag{35}$$

Regarding the demand of items, constraint (8) is replaced by the constraint:

$$\sum_{v=1}^{S+R} \sum_{f=1}^{F_v} \sum_{j=1}^{J_v} a_{ifjv} + \sum_{k=1}^{K} \sum_{v=1}^{S+R} \sum_{f=1}^{F_v} \sum_{j=1}^{J_v} n_{ik} a_{kfjv}^{K} \leq d_i, \quad \forall i. \tag{36}$$

Different from the demand constraint (8), constraint (36) is an inequality equation since it is not possible to meet the whole demand with a single cutting pattern.

Subproblem *ST1* is solved iteratively for each type of plate in stock. Using this strategy, cutting patterns are created until the demand is met or it is no longer possible to create a new cutting pattern different from those created previously. When a cutting pattern for plate v is created, it is included in the set C_v and the demand d_i is updated. Pseudo-code "Algorithm 1" describes Step 1 of the heuristic.

Algorithm 1: Creating initial cutting patterns

```
v = 1;
while v ≤ S + R do
    v^aux = 1;
    while v^aux ≤ S+R do
        if v^aux == v then
            J_{v^aux} = 1;
            e_{v^aux} = 1;
        else
            J_{v^aux} = 0;
            e_{v^aux} = 0;
        end
        v^aux = v^aux + 1;
    end
    if v ≤ S then
        leftover = 0;
    else
        leftover = 1;
    end
    while leftover ≤ 1 do
        if leftover == 0 then
            U = 1;
        else
            U = 0;
        end
        stop = FALSE;
        d = Full demand of the problem;
        while stop == FALSE do
            Solve ST1;
            if New cutting pattern is created then
                Add the new cutting pattern to C_v;
                Update d;
                if d == 0 then
                    stop = TRUE;
                end
            else
                stop = TRUE;
            end
        end
        leftover = leftover + 1;
    end
    v = v + 1
end
```

In the pseudo-code "Algorithm 1", v specifies the object for which $ST1$ will create cutting patterns. In lines 3–13, parameters J_v and e_v are set in order to limit the number of created cutting patterns according to v. The *leftover* parameter indicates if the generation of leftovers will be considered for object v. After the definition of parameters, the repetition structure in lines 27–38 solves $ST1$ iteratively until the boolean parameter *stop* is *TRUE*, which indicates that one of the following stopping criteria has been reached: (i) the demand for all types of items is met (line 32–34); (ii) or it is not possible to create a new cutting pattern (line 35–37). At the end of Step 1, the set C_v will contain sufficient good-quality cutting patterns for all types of plates in stock. These cutting patterns will allow the feasibility of Subproblem *ST2* that is solved in all iterations of Step 2 described in the next subsection, since *ST2* will create, at most, one new cutting pattern at each iteration.

5.2 Step 2: solving the adapted model

In Step 2 of the proposed heuristic, the linear model was adapted to solve the 2D-CSPUL using the cutting patterns generated in Step 1. This adaptation could also be applied to the non-linear model with few modifications.

In addition to the original parameters for the linear model, the adapted model $ST2$ needs some extra parameters from the cutting patterns in C_v. These parameters are:

- a'_{icv}: number of items i in cutting pattern c for plate $v, c \in C_v$;
- $g'_{cs} = \begin{cases} 1, & \text{if a leftover is generated in cutting pattern } c \text{ for standard plate } s, \\ & c \in C_s. \\ 0, & \text{otherwise.} \end{cases}$
- lo'_{cs}: height of leftover generated in cutting pattern c for standard plate $s, c \in C_s$.

With these parameters, it is possible to update the constraints regarding the demand, stock and generation of usable leftovers, as well the objective function. Also, a new integer decision variable is needed to indicate the frequency of each cutting pattern in C_v, and it is defined as follows:

- x'_{cv}: number of plates v cut according to cutting pattern c, $c \in C_v$;

In *ST2*, the constraints regarding the creation of cutting patterns remain the same as in the linear model, and the objective function (18) is replaced by:

$$\begin{aligned} \text{Min} \quad : & \sum_{v=1}^{S+R} \sum_{j=1}^{J_v} \sum_{\beta=1}^{e_v} \beta W_v H_v x_{\beta j v} - \alpha \sum_{s=1}^{S} \sum_{j=1}^{J_s} \sum_{\beta=1}^{e_s} \beta W_s z^O_{\beta j s} \\ & + \sum_{v=1}^{S+R} \sum_{c \in C_v} \left(W_v H_v - \alpha W_v lo'_{cv} \right) x'_{cv}. \end{aligned} \tag{37}$$

The objective function (37) differs from (18) by considering the area of both cut plates and generated leftovers obtained from the cutting patterns built in Step 1. Similar changes are made to the demand constraint (22), in which the new decision variable x'_{cv} multiplies the parameter a'_{icv}, allowing *ST2* to use these cutting patterns to meet the demand for items. The demand constraint (22) is replaced by:

$$\begin{aligned} & \sum_{v=1}^{S+R} \sum_{f=1}^{F_v} \sum_{j=1}^{J_v} \sum_{\beta=1}^{e_v} \beta z^I_{\beta i f j v} \\ & + \sum_{k=1}^{K} \sum_{v=1}^{S+R} \sum_{f=1}^{F_v} \sum_{j=1}^{J_v} \sum_{\beta=1}^{e_v} \beta n_{ik} z^K_{\beta k f j v} + \sum_{v=1}^{S+R} \sum_{c \in C_v} a'_{icv} x'_{cv} = d_i, \quad \forall i. \end{aligned} \tag{38}$$

Finally, the constraint (32) that limits the number of generated leftovers, and the stock constraint (33) are replaced, respectively, by the constraints:

$$\sum_{j=1}^{J_v} \sum_{\beta=1}^{e_v} \beta x_{\beta j v} + \sum_{v=1}^{S+R} \sum_{c \in C_v} x'_{cv} \leq e_v, \quad \forall v. \tag{39}$$

$$\sum_{s=1}^{S} \sum_{j=1}^{J_s} \sum_{\beta=1}^{e_s} \beta b_{\beta j s} + \sum_{s=1}^{S} \sum_{c \in C_s} g'_{cs} x'_{cs} \leq U. \tag{40}$$

Besides the cutting patterns in C_v, *ST2* can also build one new cutting pattern at each iteration of the procedure. As in Step 1, parameter J_v is limited to $\sum_{v=1}^{S+R} J_v = 1$ to indicate the type of plate for which the new cutting pattern is built. Pseudo-code "Algorithm 2" describes Step 2 of the heuristic procedure.

Algorithm 2: Solving *ST2*

```
v = 1;
stop = 0;
while stop < S + R do
    v^aux = 1;
    while v^aux ≤ S+R do
        if v^aux == v then
            J_{v^aux} = 1;
        else
            J_{v^aux} = 0;
        end
        v^aux = v^aux + 1;
    end
    Solve ST2;
    if Better solution found then
        Add the new cutting pattern to C_v;
        stop = 0;
    else
        stop =stop + 1;
    end
    if v == S+R then
        v = 1;
    else
        v = v + 1;
    end
end
```

In the pseudo-code "Algorithm 2", v specifies the plate for which $ST2$ is allowed to create a new cutting pattern, and *stop* is the counter variable that indicates the end of the procedure. Depending on the size of the instance, even allowing the creation of only one new cutting pattern, the solving time of $ST2$ can be long. Thus, the execution time limit is set to 600 s. After $ST2$ is solved, the procedure checks if the solution found is better than the last best solution. If it is true, the new cutting pattern that improved the solution is added to C_v, and *stop* is reset to zero. Otherwise, *stop* is incremented. By expanding the set C_v, we allow $ST2$ to improve the solution in the following iterations.

To illustrate Step 2 of the proposed heuristic procedure, we provide a numerical example. Consider an instance, whose details are given in Table 3, with $S = 2$ types of standard plates available in stock, $I = 3$ types of demanded items, and four initial cutting patterns. Table 4 presents the details of the solutions obtained at each

Table 3 Data of the numerical example

Plate type	Width × height	Stock
1	70 × 92	5
2	61 × 80	5
Item type	Width × height	Demand
1	8 × 19	59
2	9 × 10	37
3	6 × 7	49
Id	Cutting pattern	Plate type
1	(16 35 7)	1
2	(0 5 3)	1
3	(7 36 6)	2
4	(9 4 4)	2

Table 4 Solution of the numerical example

Iteration	New cutting pattern				Best solution	Best
	Plate type	Id	Items	Left.		OF value
1	1	5	(14 17 29)	0 × 0	$x_4 = 5, x_5 = 1$	30,840
2	2	6	(18 8 8)	0 × 0	$x_4 = 1, x_5 = 1, x_6 = 2$	21,080
3	1	7	(27 12 12)	0 × 0	$x_5 = 1, x_6 = 1, x_7 = 1$	17,760
4	2	x	xxxxxx	xxxxxx	$x_5 = 1, x_6 = 1, x_7 = 1$	17,760
5	1	8	(5 13 25)	70 × 43	$x_7 = 2, x_8 = 1$	16,310
6	2	x	xxxxxx	xxxxxx	$x_7 = 2, x_8 = 1$	16,310
7	1	x	xxxxxx	xxxxxx	$x_7 = 2, x_8 = 1$	16,310

iteration of the procedure. In this table, column "Plate" indicates the type of plate for which the creation of a new cutting pattern is allowed. Columns "Items" and "Left." show the number of produced items and the dimensions, width × height, of the leftover generated by the new cutting pattern. Regarding the solutions obtained at each iteration, the last two columns present the frequencies of the used cutting patterns and the value of the objective function.

In the first iteration, besides considering the four initial cutting patterns, $ST2$ was solved by being allowed to create one new cutting pattern for Plate 1. The objective function value of the obtained solution is equal to 30,840, and the solution uses the initial Pattern 4 and the new Pattern 5. This Pattern 5 was then included to set C to be considered by $ST2$ in the next iteration along with the four initial cutting patterns.

In the second iteration, $ST2$ was solved by being allowed to create one new cutting pattern for Plate 2, obtaining a better solution using the new Pattern 6 along with Patterns 4 and 5. The same happened in the third iteration when *ST*2 created a new Pattern 7 for Plate 1 and obtained a better solution with the objective function value 17,760.

In the fourth iteration, for the first time, *ST*2 could not improve the current best solution, even being allowed to create a new cutting pattern for Plate 2. But in the fifth iteration, *ST*2 created the Pattern 8 for Plate 1 and obtained a better solution with the objective function value 16,310. Pattern 8 was the first that generated a usable leftover. However, *ST*2 could not create a new cutting pattern that improves the current best solution in the sixth and seventh iterations for Plates 1 and 2, which indicates the heuristic execution must stop.

6 Computational tests

In this section, we present the description and the results of the computational tests. Section 6.1 presents the results for small-sized instances from the literature, while in Sect. 6.2 we used randomly generated medium-sized instances.

To evaluate the performance of the proposed model and heuristic procedure, they were compared to the models proposed by Andrade et al. (2016) and Furini et al. (2016). All these models were coded using OPL (Optimization Programming Language) with the CPLEX, version 12.10 solver. The computational tests were run on an Intel Core i7, 2.8 GHz, 16 GB RAM computer, with Windows 10 operating system.

The results presented in this section refer only to the linearized model proposed here. Preliminary tests with the original non-linear model presented in Sect. 3.1 produced poor results, which led us to use the linearization strategy presented in Sect. 4.

6.1 Computational results for instances from the literature

Andrade et al. (2016) proposed two mathematical models, which are referred to here as M_1 and M_2, to solve the 2-staged 2D-CSPUL. The only difference between the models is regarding the demand for items. In M_1, all items have a demand equal to 1, as in the bin-packing problem. In M_2, items of the same type are grouped together. The objective function of both models is to minimize the total area of the cut plates and, among the minimum value solutions, choose the one that maximizes the area of usable leftovers. For a fair comparison, in the computational tests presented in this subsection, the objective function (18) described in Sect. 4 was adapted to be equivalent to the objective function of the models M_1 and M_2. The adapted objective function is defined as follows:

$$\text{Min} \ : \sum_{s=1}^{S} \sum_{j=1}^{J_s} \sum_{\beta=1}^{e_s} \beta W_s H_s x_{\beta js} - \left(\sum_{s=1}^{S} \sum_{j=1}^{J_s} \sum_{\beta=1}^{e_s} \beta W_s z^{O}_{\beta js} \right) / \left(\sum_{s=1}^{S} W_s H_s e_s \right) \quad (41)$$

Table 5 Description of the instances from Andrade et al. (2016)

Instance	S	v	I	I'
1	1	3	32	32
2	1	2	23	23
3	4	7	17	17
4	4	11	27	27
5	4	16	37	37
6	3	8	34	34
7	3	3	9	11
8	3	10	19	19
9	2	4	17	17
10	3	6	10	13
11	3	13	5	37
12	1	1	1	12
13	2	7	2	32
14	3	4	3	28
15	3	12	3	34
16	3	19	3	21
17	1	1	2	17
18	2	5	2	24
19	3	16	3	41
20	2	9	3	21

The computational tests were performed with the same 20 instances presented by Andrade et al. (2016). Table 5 presents a short description of the instances. For the complete description, see the Appendix. Table 5 shows the number of types of plates in stock (S), the total number of plates (v), the number of types of ordered items (I) and the total number of items (I') for each instance.

As the models M_1 and M_2 consider the 2-staged 2D-CSPUL, the instances were solved without considering macro-items. In the case of the model (1)–(17), two scenarios were tested for the maximum number of generated leftovers, $U = 1$ and $U = 5$, and the parameter α was set to 1 for both scenarios. As in Andrade et al. (2016), the minimum height of generated leftovers was equal to the height of the smallest item and there was no limitation for the maximum height of generated leftovers. The computational time limit (TL) allowed for all models was 3600 s.

Tables 6 and 7 show the results obtained by the models and the heuristic procedure, respectively, for the instances from Andrade et al. (2016). All models found the optimal solution for all instances, with only the computational time varying between them. The total area of cut plates (obj.), the total area of generated leftovers (left.), the value for the objective function (OF) and the solving time for M_1 ($Time^{M_1}$), M_2 ($Time^{M_2}$), scenario $U = 1$ ($Time^{U=1}$) and scenario $U = 5$ ($Time^{U=5}$) are given. In Table 6, the shortest solving time for each instance is highlighted in bold, as are the instances that reached the optimal solution in Tables 7, 8, 9, 10 and 11.

Table 6 Results for the 2-staged models

Inst.	Obj.	Left.	OF	$Time^{M_1}$	$Time^{M_2}$	$Time^{U=1}$	$Time^{U=5}$
1	5512	520	5511.93	89.14	92.49	**2.44**	2.81
2	7560	2898	7559.62	0.27	0.30	**0.23**	0.36
3	260	52	259.96	**0.92**	0.98	1.41	2.30
4	360	0	360.00	522.00	613.00	**10.64**	47.84
5	466	0	466.00	67.61	36.63	**33.69**	91.56
6	492	48	491.98	16.97	17.03	14.83	**4.94**
7	180	108	179.87	0.06	**0.05**	**0.05**	0.08
8	864	64	863.98	**1.39**	1.67	9.72	45.59
9	380	0	380.00	**0.14**	0.17	0.17	**0.14**
10	51,216	12,998	51,215.92	0.19	**0.17**	0.55	0.95
11	1746	60	1745.99	TL	558.13	2.39	**1.36**
12	266	154	265.42	**0.01**	0.02	**0.01**	**0.01**
13	684	100	683.95	3.55	0.34	0.14	**0.13**
14	180	18	179.98	0.44	0.11	**0.09**	**0.09**
15	1506	0	1506.00	TL	2.67	2.52	**2.20**
16	1365	36	1364.99	2.20	**0.97**	1.30	1.38
17	266	168	265.37	**0.01**	0.02	**0.01**	**0.01**
18	748	0	748.00	0.17	0.09	**0.08**	0.09
19	2010	0	2010.00	9.89	1.42	**0.64**	1.13
20	1168	132	1167.96	2.23	0.45	**0.11**	0.20

TL = 3600 s

Although the solving time varied considerably between the models, we can see that the model proposed in this paper was significantly faster than M_1 and M_2, both for $U = 1$ and $U = 5$, in three instances (1, 4 and 11), and considerably slower for instance 8. For the other instances, the solving times were similar despite M_1 reaching the maximum execution time for two instances, 11 and 15. Regarding the results obtained by the heuristic procedure, the optimal solution was found for 10 of the 20 instances. However, for all those 10 instances, the solving time was longer than all the tested models because of the time spent with the creation of the initial cutting patterns for all types of objects in Step 1 of the procedure, although the solving times for Step 2 were very similar to those spent by the other two models.

The instances from Andrade et al. (2016) were also solved considering macro-items. All possible macro-items were given to M_2 as new items, and the model was modified to include the parameter n_{ik} in the demand constraint, which indicates the number of items i produced from macro-item k. This modification could not be made in M_1 since each macro-item could only be used once, its demand being limited to 1. An alternative to overcome this limitation would be to include each macro-item repeatedly as an input parameter. However, the number of times each macro-item would be used to reach optimality is not known.

Table 7 Results for the heuristic procedure without macro-items

Inst.	$U = 1$				$U = 5$			
	Obj.	Left.	OF	$Time^{U=1}$	Obj.	Left.	OF	$Time^{U=5}$
1	5512	468	5511.94	11.25	5512	468	5511.94	11.23
2	7560	2772	7559.63	6.74	7560	2835	7559.62	6.69
3	260	39	259.97	8.05	260	39	259.97	6.43
4	390	39	389.98	12.34	390	39	389.98	10.94
5	490	13	489.99	32.26	490	13	489.99	31.27
6	492	48	**491.98**	30.29	492	48	**491.98**	30.51
7	180	108	**179.87**	15.42	180	108	**179.87**	14.98
8	864	48	863.99	13.78	864	48	863.99	14.42
9	494	114	493.89	8.84	494	114	493.89	8.89
10	51,216	12,028	51,215.93	12.00	51,216	12,028	51,215.93	12.34
11	1746	54	1745.99	19.61	1746	54	1745.99	18.52
12	266	154	**265.42**	0.87	266	154	**265.42**	0.87
13	684	100	**683.95**	5.62	684	100	**683.95**	5.75
14	180	18	**179.98**	3.51	180	18	**179.98**	3.75
15	1506	0	**1506.00**	25.39	1506	0	**1506.00**	25.88
16	1365	36	**1364.99**	10.76	1365	36	**1364.99**	9.41
17	266	168	**265.37**	1.57	266	168	**265.37**	1.57
18	748	0	**748.00**	19.16	748	0	**748.00**	19.24
19	2010	0	**2010.00**	9.93	2010	0	**2010.00**	9.06
20	1214	0	1214.00	17.72	1168	0	1168.00	17.94

Due to certain characteristics of the instances from Andrade et al. (2016), mainly the wide variety of items, some instances have a huge number of possible macro-items. Thus, we tested two scenarios: (i) considering all the possible macro-items; and (ii) limiting the use of macro-items to only those whose constructed rectangle generates a maximum of 10% of waste. Tables 8, 9, 10 and 11 show the results obtained by the models and the heuristic procedure for these two scenarios. All the column headings are the same as in Tables 6 and 7, except for a new "K" column that shows the number of types of macro-items for each instance, and a new "Macro (%)" column that shows the percentage of macro-items used in the solutions. The results obtained by M_2 and the model proposed in this paper for $U = 5$ are shown together (M_2 / $U = 5$) in Table 8 because the solutions were the same. In Tables 8 and 10, the best solutions for each instance and their respective lowest solving time are highlighted in bold. And the same was done in Tables 9 and 11 for those instances in which the heuristic procedure reached the optimal solution.

In Tables 8 and 10, Instances 12, 13, 14, 17 and 19 have the same values in all columns as in Table 6 because there are no possible macro-items for these instances. And Instances 15, 16, 18 and 20 have the same values in all columns of Table 8 and

Table 8 Results for the 3-staged models with limited macro-items

Inst.	K	$U = 1$					M_2 / $U = 5$					
		Obj.	Left.	Macro (%)	OF	$Time^{U=1}$	Obj.	Left.	Macro (%)	OF	$Time^{M_2}$	$Time^{U=5}$
1	67	5512	780	33.33	**5511.91**	**11.31**	5512	780	39.13	**5511.91**	TL	15.47
2	42	7560	2898	21.05	7559.62	0.88	7560	3087	21.05	**7559.59**	142.25	**4.47**
3	32	180	0	41.67	**180.00**	1.66	180	0	33.33	**180.00**	**1.55**	2.67
4	90	336	0	31.58	**336.00**	56.97	336	0	56.25	**336.00**	**47.47**	108.66
5	128	466	13	37.04	**465.99**	**1500.14**	466	13	15.15	**465.99**	TL	TL
6	107	492	72	37.50	**491.96**	**37.23**	492	72	25.93	**491.96**	2015.25	140.64
7	2	180	108	0.00	**179.87**	0.06	180	108	0.00	**179.87**	**0.05**	0.06
8	20	775	0	11.76	**775.00**	18.09	775	0	11.76	**775.00**	**3.06**	16.66
9	41	380	38	50.00	379.96	0.22	380	57	50.00	**379.95**	2.84	**0.27**
10	4	51,216	14,938	9.09	**51,215.91**	0.72	51,216	14,938	9.09	**51,215.91**	**0.20**	2.13
11	2	1746	120	9.09	**1745.98**	**5.97**	1746	120	9.09	**1745.98**	114.30	10.92
12	0	266	154	0.00	**265.42**	**0.01**	266	154	0.00	**265.42**	0.02	**0.01**
13	0	684	100	0.00	**683.95**	0.14	684	100	0.00	**683.95**	0.34	**0.13**
14	0	180	18	0.00	**179.98**	**0.09**	180	18	0.00	**179.98**	0.11	**0.09**
15	1	1494	0	17.24	**1494.00**	**1.86**	1494	0	17.24	**1494.00**	10.45	4.30
16	7	1362	0	18.18	**1362.00**	14.99	1362	0	18.18	**1362.00**	**1.34**	13.00
17	0	266	168	0.00	**265.37**	**0.01**	266	168	0.00	**265.37**	0.02	**0.01**
18	1	748	132	14.29	**747.93**	0.19	748	132	14.29	**747.93**	0.14	**0.09**
19	0	2010	0	0.00	**2010.00**	**0.64**	2010	0	0.00	**2010.00**	1.42	1.13
20	1	1168	132	5.00	**1167.96**	**0.16**	1168	132	5.00	**1167.96**	0.47	0.27

TL = 3600 s

Table 9 Results for the heuristic procedure with limited macro-items

Inst.	$U = 1$					$U = 5$				
	Obj.	Left.	Macro (%)	OF	$Time^{U=1}$	Obj.	Left.	Macro (%)	OF	$Time^{U=5}$
1	5512	572	23.08	5511.93	9.87	5512	572	23.08	5511.93	9.59
2	7560	2898	21.05	**7559.62**	3.20	7560	3087	21.05	**7559.59**	4.89
3	180	0	33.33	**180.00**	7.95	180	0	33.33	**180.00**	7.99
4	336	0	50.00	**336.00**	33.94	336	0	30.00	**336.00**	36.54
5	466	13	5.71	**465.99**	59.18	466	13	8.82	**465.99**	56.66
6	492	72	17.86	**491.96**	27.05	492	72	17.86	**491.96**	27.62
7	180	108	0.00	**179.87**	3.73	180	108	0.00	**179.87**	4.09
8	775	0	26.67	**775.00**	7.91	775	0	26.67	**775.00**	7.43
9	380	38	40.00	**379.96**	3.10	380	57	40.00	**379.95**	2.52
10	51,216	14,938	9.09	**51,215.91**	4.59	51,216	14,938	9.09	**51,215.91**	5.08
11	1746	60	12.50	1745.99	7.66	1746	60	20.83	1745.99	6.60
12	266	154	0.00	**265.42**	0.87	266	154	0.00	**265.42**	0.87
13	684	100	0.00	**683.95**	5.62	684	100	0.00	**683.95**	5.75
14	180	18	0.00	**179.98**	3.51	180	18	0.00	**179.98**	3.75
15	1506	0	0.00	1506.00	8.29	1506	0	0.00	1506.00	8.91
16	1362	0	18.18	**1362.00**	5.12	1362	0	18.18	**1362.00**	4.41
17	266	168	0.00	**265.37**	1.57	266	168	0.00	**265.37**	1.57
18	748	132	14.29	**747.93**	3.75	748	132	14.29	**747.93**	3.85
19	2010	0	0.00	**2010.00**	9.93	2010	0	0.00	**2010.00**	9.06
20	1214	132	10.53	1213.96	5.18	1214	132	10.53	1213.96	4.71

Table 10 Results for the 3-staged models with unlimited macro-items

Inst.	K	M_2				$U=5$				
		Obj.	Left.	OF	$Time^{M_2}$	Obj.	Left.	Macro (%)	OF	$Time^{U=5}$
1	818	5512	780	**5511.91**	TL	5512	832	33.33	**5511.90**	**607.56**
2	249	7560	3087	**7559.59**	TL	7560	3087	21.05	**7559.59**	**1105.02**
3	350	180	0	**180.00**	257.53	180	0	41.67	**180.00**	**36.72**
4	952	*	*	*	*	336	0	50.00	**336.00**	**1641.14**
5	2171	*	*	*	*	*	*	*	*	*
6	1554	*	*	*	*	492	72	43.48	**491.96**	**937.73**
7	11	180	108	**179.87**	0.16	180	108	0.00	**179.87**	**0.08**
8	70	775	26	**774.99**	40.55	775	26	26.67	**774.99**	**39.20**
9	291	380	76	**379.93**	167.66	380	76	50.00	**379.93**	**1.39**
10	19	51,216	14,938	**51,215.91**	**0.73**	51,216	14,938	9.09	**51,215.91**	10.48
11	5	1746	120	**1745.98**	369.36	1746	120	9.09	**1745.98**	**65.86**
12	0	266	154	**265.42**	0.02	266	154	0.00	**265.42**	**0.01**
13	0	684	100	**683.95**	0.34	684	100	0.00	**683.95**	**0.13**
14	0	180	18	**179.98**	0.11	180	18	0.00	**179.98**	**0.09**
15	1	1494	0	**1494.00**	10.45	1494	0	17.24	**1494.00**	**4.30**
16	7	1362	0	**1362.00**	**1.34**	1362	0	18.18	**1362.00**	13.00
17	0	266	168	**265.37**	0.02	266	168	0.00	**265.37**	**0.01**
18	1	748	132	**747.93**	0.14	748	132	14.29	**747.93**	**0.09**
19	0	2010	0	**2010.00**	1.42	2010	0	0.00	**2010.00**	**1.13**
20	1	1168	132	**1167.96**	0.47	1168	132	5.00	**1167.96**	**0.27**

TL = 3600 s

also in Table 10, since their possible macro-items generate a waste lower than 10% in the constructed rectangle.

In the scenario with limited macro-items, the proposed model found the optimal solution obtained by M_2 for all instances only when $U = 5$. With $U = 1$, the model found worst solutions for Instances 2 and 9. In comparison with the 2-staged problem, using macro-items allowed improvement in the solution of thirteen out of fifteen instances. And for two of them, instances 4 and 9, solutions were found composed of at least 50% of macro-items.

Regarding the solving time, the model with $U = 5$ was considerably faster than M_2 for five instances (1, 2, 6, 11 and 15) and slower for Instances 4, 8 and 16. For the remaining instances, the solving time was similar, including when $U = 1$. Both models reached the maximum execution time for Instance 5, which has 128 macro-items, the highest number generated among the instances.

The increase in the number of possible macro-items allowed an improvement in the solution of two instances, 8 and 9, maintaining a low solving time, as seen in Table 10. With unlimited macro-items, the solver ran out of memory without

Table 11 Results for the heuristic procedure with unlimited macro-items

Inst.	$U = 1$					$U = 5$				
	Obj.	Left.	Macro (%)	OF	$Time^{U=1}$	Obj.	Left.	Macro (%)	OF	$Time^{U=5}$
1	5512	676	19.23	5511.92	72.45	5512	676	14.81	5511.92	86.71
2	7560	2898	21.05	7559.62	9.15	7560	3087	15.79	**7559.59**	13.73
3	180	0	33.33	**180.00**	41.00	180	0	33.33	**180.00**	46.81
4	336	0	73.33	**336.00**	331.04	336	0	42.11	**336.00**	373.17
5	466	13	12.12	465.99	1233.50	466	13	20.00	465.99	1322.47
6	492	54	14.29	491.97	449.96	492	54	22.22	491.97	450.77
7	180	108	0.00	**179.87**	9.79	180	108	0.00	**179.87**	10.15
8	775	0	26.67	775.00	14.11	775	0	26.67	775.00	13.26
9	380	38	40.00	379.96	8.69	380	57	40.00	379.95	8.32
10	51,216	14,938	9.09	**51,215.91**	7.40	51,216	14,938	9.09	**51,215.91**	7.94
11	1746	60	17.39	1745.99	12.51	1746	60	17.39	1745.99	10.98
12	266	154	0.00	**265.42**	0.87	266	154	0.00	**265.42**	0.87
13	684	100	0.00	**683.95**	5.62	684	100	0.00	**683.95**	5.75
14	180	18	0.00	**179.98**	3.51	180	18	0.00	**179.98**	3.75
15	1506	0	0.00	1506.00	8.29	1506	0	0.00	1506.00	8.91
16	1362	0	18.18	**1362.00**	5.12	1362	0	18.18	**1362.00**	4.41
17	266	168	0.00	**265.37**	1.57	266	168	0.00	**265.37**	1.57
18	748	132	14.29	**747.93**	3.75	748	132	14.29	**747.93**	3.85
19	2010	0	0.00	**2010.00**	9.93	2010	0	0.00	**2010.00**	9.06
20	1214	132	10.53	1213.96	5.18	1214	132	10.53	1213.96	4.71

finding a feasible solution after a few minutes of execution of M_2 for Instances 4, 5 and 6. The same happened to the model proposed in this paper but only for instance 5. This occurred due to the huge number of possible macro-items in these instances. Also, M_2 reached the maximum execution time for instances 1 and 2, finding a worse solution than the solution found by the model proposed in this paper for Instance 1. The heuristic procedure found the optimal solution for 80% of the instances with limited macro-items and 60% with unlimited macro-items. The procedure considerably reduced the solving time for the first six instances, especially Instance 5 with unlimited macro-items, for which none of the models found a solution, while the heuristic solving time was equal to 1322.47 s.

6.2 Computational results for randomly generated instances

The main objective of the tests is to analyze the behavior of the proposed model and heuristic in a scenario with a higher number of items per instance.

16 classes of instances were defined with $S = 2$ types of standard plates and varying the number of types of leftovers in stock ($R = 1$ and 2). The maximum number of cutting patterns for each plate v (J_v) was defined as equal to the number of plates v available in stock. We considered $I = 3, 5, 8$, and 10 types of items. Their dimensions were generated according to the dimensions of the plates and leftovers in stock. Two scenarios were considered for the dimensions of items: (i) height and width generated in the intervals $[0.05 \times H^{max}, 0.25 \times H^{max}]$ and $[0.05 \times W^{max}, 0.25 \times W^{max}]$, respectively, and (ii) height and width generated in the intervals

Table 12 Description of the classes of instances

Class	S	R	Items size	I	v	I'	K'
1	2	1	5–25%	3	7.6	86.3	1.9
2	2	1	5–25%	5	8.4	127.4	3.9
3	2	1	5–25%	8	9.0	171.3	10.6
4	2	1	5–25%	10	14.0	239.6	12.3
5	2	1	10–50%	3	14.8	95.6	2.5
6	2	1	10–50%	5	15.6	118.4	4.2
7	2	1	10–50%	8	22.1	191.7	6.1
8	2	1	10–50%	10	23.8	230.2	8.1
9	2	2	5–25%	3	9.9	97.8	3.5
10	2	2	5–25%	5	9.5	130.4	4.7
11	2	2	5–25%	8	12.4	182.8	8.9
12	2	2	5–25%	10	14.4	242.6	10.7
13	2	2	10–50%	3	15.4	91.4	2.3
14	2	2	10–50%	5	15.8	121.8	4.6
15	2	2	10–50%	8	21.4	185.2	7.9
16	2	2	10–50%	10	27.3	222.6	13.0

$[0.1 \times H^{max}, 0.5 \times H^{max}]$ and $[0.1 \times W^{max}, 0.5 \times W^{max}]$, where H^{max} is the maximum height among the standard plates and W^{max} is the maximum width among the standard plates. The number of plates in stock was generated in order to ensure that there was at least one unit of each type of standard plate and each type of leftover. In addition, the total area of plates in stock was always greater than 120% of the total area of ordered items. Table 12 describes the classes. For each class, 10 instances were generated, and the last three columns of Table 12 show the average values of plates in stock (v), ordered items (I'), and possible macro-items (K'). The full description of all instances can be found at http://data.mendeley.com/datasets/ddc79swng4/1.

Initially, all instances were solved without the generation of leftovers ($U = 0$). The results obtained by the model and the heuristic procedure were compared with the results obtained by the model proposed in Furini et al. (2016). In that paper, the authors addressed mainly the guillotine two-dimensional knapsack problem, but also presented extensions to other problems, including the guillotine two-dimensional cutting stock problem (2D-CSP). The strategy used by the model is to cut the original plates in stock through horizontal and vertical guillotine cuts. The resulting plates can also be cut or kept to satisfy the demand if their dimensions are equal to the dimensions of one of the ordered items. This process is repeated until the resulting plates are larger than one of the ordered items. Therefore, the model proposed by Furini et al. (2016) does not limit the number of cutting stages. A procedure is proposed to enumerate the set of possible plates obtained through the guillotine horizontal and vertical cuts, as well as their related parameters and variables.

Regarding the computational tests for the classes described in Table 12, the procedure proposed by Furini et al. (2016) was adapted to limit the problem to three stages, considered by the proposed model and heuristic procedure to the 2D-CSPUL. Thus, both models and the heuristic procedure have the same solution space. Table 13 shows the average results of the model proposed by Furini et al. (2016), and the model and heuristic procedure proposed in this paper for all classes of instances with $U = 0$. The average value for the objective function (OF), the average computational time in s (time (s)) and the average gap (gap (%)) are given. This gap is a parameter provided by the CPLEX solver that indicates the distance between the current best integer solution and the best node remaining. If the average gap of a class is greater than 0, it means that for at least one instance the maximum solving time was reached. Table 13 also shows the average number of initial cutting patterns (Init.) and the average number of new cutting patterns (new) generated by the heuristic procedure.

The proposed model presented better results for the classes with fewer types of items, mainly Classes 2 and 9. For these classes, the average value for the objective function was the same as the adapted model from Furini et al. (2016) but with a gap equal to 0 and an average solving time 2000 s faster, for Class 2, and 390 s faster for Class 9. For the classes with larger items and more types of items, the proposed model had higher average gaps, around 60% and 70%. The heuristic procedure, however, was more efficient in all classes with $I = 8$ and $I = 10$ types of items,

Table 13 Results for the instances randomly generated with $U = 0$

Class	Furini et al. (2016)			Model			Heuristic			
	OF	Time (s)	Gap (%)	OF	Time (s)	Gap (%)	OF	Time (s)	Init.	New
1	16,085.10	10.25	0.00	16,085.10	3.21	0.00	16,150.10	4.43	29.60	1.90
2	27,245.50	2331.21	2.07	27,245.50	322.49	0.00	27,547.60	11.07	41.70	1.60
3	27,208.20	TL	2.92	27,738.50	2526.90	7.96	27,691.50	37.36	45.40	1.30
4	44,761.10	TL	2.78	47,134.50	3249.70	30.46	45,215.80	621.36	68.90	1.40
5	67,787.00	0.66	0.00	67,787.00	1436.12	0.33	68,095.80	21.02	92.70	2.00
6	72,443.10	2897.23	0.98	72,450.00	3190.10	7.18	72,736.40	61.42	102.40	2.50
7	106,331.70	2507.45	0.40	110,668.90	TL	34.55	107,969.60	813.86	151.10	3.20
8	117,978.40	TL	0.64	132,096.50	TL	60.81	120,527.00	1633.91	159.60	3.60
9	15,931.10	395.62	0.32	15,931.10	6.95	0.00	16,067.20	9.33	37.00	1.80
10	18,483.30	1986.78	1.08	18,483.30	796.92	0.22	18,660.20	21.25	43.90	1.60
11	26,242.00	3171.32	1.10	27,204.80	2999.20	14.08	26,844.50	116.84	69.10	1.40
12	38,760.50	TL	1.04	41,229.40	TL	25.38	39,553.40	394.98	85.70	2.00
13	67,374.10	4.42	0.00	67,465.90	674.71	1.47	67,904.80	20.01	89.70	2.00
14	64,916.90	1836.18	0.28	65,113.30	3189.80	10.71	65,262.80	55.32	115.20	2.70
15	93,915.00	2737.60	0.30	97,850.90	TL	34.62	94,991.80	529.13	173.00	4.80
16	113,582.60	3064.57	0.37	128,084.80	TL	71.03	115,269.90	1173.05	197.40	4.80

TL = 3600 s

Table 14 Results for the instances randomly generated with $U = 1$ and $\alpha = 100\%$

Class	Model				Heuristic				
	OF	Left.	Time (s)	Gap (%)	OF	Left.	Time (s)	Init.	New
1	15,750.10	0.90	10.22	0.00	15,854.30	0.80	4.53	29.60	1.60
2	26,689.80	0.90	2011.74	0.38	27,053.00	0.70	12.88	41.70	1.60
3	27,338.60	0.70	3594.63	10.43	27,149.30	0.80	45.80	45.40	1.20
4	46,719.50	0.60	3259.53	31.77	44,768.90	1.00	750.13	68.90	1.60
5	67,216.30	0.70	678.11	0.23	67,216.30	0.70	15.45	92.70	2.60
6	71,955.60	0.90	3358.50	7.77	71,693.80	1.00	110.56	102.40	3.30
7	110,352.80	0.00	TL	35.00	106,761.40	0.90	837.73	151.10	3.90
8	129,242.30	0.10	TL	58.80	119,258.80	1.00	1515.66	159.60	4.40
9	15,708.80	1.00	116.65	0.00	15,816.80	1.00	7.73	37.00	2.60
10	18,007.80	1.00	1347.16	0.65	18,152.70	1.00	24.12	43.90	2.20
11	26,770.40	0.90	TL	16.85	26,639.30	0.90	180.50	69.10	1.90
12	40,578.10	0.80	TL	29.26	39,202.40	0.80	468.19	85.70	1.80
13	67,032.90	0.70	826.15	1.27	67,546.20	0.70	24.14	89.70	2.00
14	64,682.90	0.90	TL	9.65	64,839.30	1.00	41.17	115.20	2.70
15	96,884.10	0.20	TL	36.50	94,515.80	1.00	377.54	173.00	4.20
16	128,109.60	0.00	TL	74.51	114,345.60	0.90	817.30	197.40	4.80

TL = 3600 s

Table 15 Results for the instances randomly generated with $U = 1$ and $\alpha = 90\%$

Class	Model				Heuristic				
	OF	Left.	Time (s)	Gap (%)	OF	Left.	Time (s)	Init.	New
1	15,890.10	0.40	10.39	0.00	15,981.50	0.40	4.22	29.60	2.00
2	26,906.28	0.80	1194.04	0.07	27,379.36	0.60	14.93	41.70	1.50
3	27,633.50	0.60	TL	11.81	27,367.00	0.60	45.52	45.40	1.50
4	47,255.50	0.20	3255.89	37.35	44,950.52	0.50	944.53	68.90	1.80
5	67,384.80	0.40	703.87	0.14	67,384.80	0.40	14.69	92.70	2.50
6	72,408.75	0.60	3416.30	7.78	72,108.20	0.80	131.96	102.40	3.50
7	110,235.60	0.10	TL	33.98	107,123.62	0.80	898.87	151.10	3.80
8	130,665.70	0.10	TL	54.39	119,462.45	0.90	1589.34	159.60	4.50
9	15,884.80	0.40	44.60	0.00	15,995.20	0.20	7.57	37.00	2.00
10	18,233.80	0.60	1469.52	1.09	18,415.42	0.50	18.75	43.90	2.30
11	26,936.80	0.50	TL	15.02	26,737.20	0.10	114.27	69.10	1.60
12	41,078.10	0.40	TL	30.39	39,400.40	0.40	527.56	85.70	1.70
13	67,229.81	0.50	793.02	1.23	67,645.80	0.30	21.05	89.70	2.30
14	64,995.77	0.40	TL	8.93	64,881.80	0.40	46.24	115.20	3.60
15	97,083.54	0.30	TL	32.04	94,751.94	0.60	426.20	173.00	4.00
16	125,895.80	0.00	TL	69.65	114,527.90	0.30	1037.16	197.40	4.50

TL = 3600 s

Table 16 Results for the instances randomly generated with $U = 5$ and $\alpha = 100\%$

Class	Model				Heuristic				
	OF	Left.	Time (s)	Gap (%)	OF	Left.	Time (s)	Init.	New
1	15,321.10	3.40	138.50	0.00	15,419.40	3.90	6.05	29.60	1.90
2	26,287.60	4.10	2275.00	2.33	26,533.40	3.80	20.32	41.70	2.00
3	26,923.70	4.10	TL	21.84	26,767.10	3.80	80.70	45.40	1.50
4	46,579.20	2.90	TL	48.95	44,308.90	4.50	1453.57	68.90	2.50
5	66,501.70	3.20	1231.50	1.24	66,570.60	3.10	19.40	92.70	2.70
6	70,795.30	4.20	TL	13.29	70,534.80	4.20	150.70	102.40	3.70
7	109,642.50	3.00	TL	47.54	105,495.00	4.60	602.48	151.10	4.20
8	131,999.50	0.80	TL	70.73	117,720.20	4.70	1533.03	159.60	6.00
9	15,507.20	4.10	461.80	0.01	15,584.60	3.40	7.81	37.00	1.50
10	17,780.50	3.70	2232.70	3.38	17,933.10	3.30	35.81	43.90	1.90
11	26,552.10	3.50	TL	25.66	26,304.90	3.50	673.29	69.10	3.40
12	39,902.90	3.20	TL	39.60	38,921.40	3.60	1153.86	85.70	2.90
13	66,524.30	2.60	1022.81	1.89	66,776.80	2.30	18.12	89.70	2.60
14	64,328.10	4.00	TL	15.14	64,103.30	3.40	49.15	115.20	3.00
15	96,450.10	3.80	TL	43.45	93,335.60	4.70	368.93	173.00	5.60
16	128,725.30	1.30	TL	76.93	113,512.30	4.90	1786.69	197.40	6.80

TL = 3600 s

Table 17 Results for the instances randomly generated with $U = 5$ and $\alpha = 90\%$

Class	Model				Heuristic				
	OF	Left.	Time (s)	Gap (%)	OF	Left.	Time (s)	Init.	New
1	15,801.16	0.90	11.20	0.00	15,872.73	0.60	4.95	29.60	1.90
2	26,875.28	1.10	2391.31	1.13	27,191.28	0.90	20.44	41.70	2.00
3	27,692.50	1.20	TL	19.49	27,300.50	0.80	60.30	45.40	1.60
4	47,209.50	1.30	3306.05	44.98	44,918.52	0.50	779.43	68.90	2.00
5	67,120.47	1.30	1494.22	0.37	67,284.80	1.10	15.19	92.70	2.10
6	71,827.85	2.30	TL	12.99	71,656.40	2.30	117.28	102.40	3.10
7	109,738.14	1.50	TL	36.34	106,388.10	2.60	523.51	151.10	3.60
8	133,000.46	1.00	TL	71.17	118,908.24	1.50	1682.13	159.60	5.40
9	15,873.68	0.50	82.82	0.00	15,973.36	0.30	7.71	37.00	2.10
10	18,229.80	0.60	1627.08	3.42	18,411.42	0.70	22.08	43.90	2.10
11	27,144.83	1.20	TL	23.01	26,714.20	0.30	251.40	69.10	1.90
12	40,781.40	1.10	TL	42.46	39,324.40	0.30	751.18	85.70	2.20
13	67,118.21	1.30	973.78	2.07	67,556.80	0.70	18.38	89.70	1.70
14	65,027.77	0.70	TL	14.51	64,874.80	0.80	49.58	115.20	3.80
15	97,169.28	1.80	TL	39.64	94,358.06	1.50	240.31	173.00	4.80
16	130,519.48	0.60	3580.52	70.62	114,477.90	0.20	956.63	197.40	4.80

TL = 3600 s

mainly for Class 16, in which the heuristic procedure found an average value of the objective function equal to 115,269.9 (Model = 128,084.8). In comparison with the adapted model from Furini et al. (2016), the solutions obtained by the heuristic procedure were very satisfactory, since they were, at most, only 2.5% worse.

The randomly generated classes of instances were also solved considering the generation of leftovers. Four scenarios were tested, varying the maximum number of generated leftovers as $U = 1$ and 5, and varying the re-use rate as $\alpha = 100\%$ and 90%. For all scenarios, we defined the minimum and maximum height for generated leftovers as $h^R_{min} = 0.4$ and $h^R_{max} = 0.6$.

Tables 14, 15, 16 and 17 show the average results of the proposed model and heuristic procedure for these scenarios. All the column headings of the tables are the same as in Table 13, except for a new column, “Left.", that shows the average number of generated leftovers.

The proposed model results had many characteristics similar to the results without the generation of leftovers. The best results were obtained for classes with $I = 3$ and 5 types of items, being better than the results obtained by the heuristic for all scenarios. However, for classes with $I = 8$ and 10, the gaps were very high, especially for classes with larger items. That occurred because, with larger items, the total number of plates in stock increases, impacting the maximum number of different cutting patterns that can be generated for each plate (J_s), making the problem computer-time costly. The heuristic procedure performed better in the tests for these classes because the higher number of plates in stock does not interfere with the number of initial cutting patterns created in Step 1 of the procedure. The heuristic procedure also improved the solutions for all classes when it was allowed to generate more leftovers. On the other hand, the model reached the maximum execution time in practically all instances of classes 8 and 16 and had better solutions for $U = 1$ than for $U = 5$. For these classes, heuristic procedure improved the solutions by up to 14% in less than half the average time.

7 Conclusions

This paper presents a new mathematical model and a heuristic procedure to solve the two-dimensional cutting stock problem with usable leftovers (2D-CSPUL). The proposed model aims to create cutting patterns and determine their respective frequencies at same time, for multiple types of plates in stock (standard plates and leftovers), allowing the generation of usable leftovers that can be cut in future cutting processes.

The proposed model considers the concept of strips, that consists of, first, dividing the plate in horizontal strips, in which multiple items can be allocated side by side since their heights does not exceed the height of the strip. Items can be combined in order to create macro-items, that also can be allocated in the strips. Leftovers are generated through a single guillotine horizontal cut in the object, and their heights must be within a predefined interval.

A heuristic procedure was proposed to provide real-world integer solutions for large problems. The strategy of this procedure is to solve the proposed model allowing the generation of only one new cutting pattern at each iteration. In order to ensure the feasibility of the procedure, a set of initial cutting patterns is created, prioritizing those with the minimum waste.

Computational tests were run to evaluate the proposed model and heuristic procedure in comparison with other models from the literature. These models were adapted and implemented to consider macro-items, and they were tested with their own instances and instances randomly generated for this paper. The proposed model was very efficient in solving the 20 instances from Andrade et al. (2016), finding the optimal solutions in reasonable computational time. On the other hand, the heuristic procedure did not perform well for these instances since they are small, with less than 50 items. In the second set of tests, with randomly generated instances, the proposed model and heuristic procedure were compared to an adapted model from Furini et al. (2016). The proposed model had better results for the classes of instances with few types of items and smaller items. For the classes with larger items and more different types, the heuristic procedure was superior and had results closer to the results of the model from Furini et al. (2016).

For future research, 2D-CSPUL extension to a multiperiod problem considering a planning horizon could be explored, or its extension to stochastic scenarios, such as where there is uncertainty in the items demand. Regarding the proposed heuristic procedure, it would be interesting to implement alternative methods to generate the set of initial patterns aiming to verify the impact on the heuristic convergence.

Appendix: Instances from Andrade et al. (2016)

See Table 18.

Table 18 Description of the instances from Andrade et al. (2016)

Inst.	Objects		Items	
	S	Width $\times$ Height	I'	Width $\times$ Height
1	3	3 (52 $\times$ 53)	32	19 $\times$ 21, 7 $\times$ 20, 4 $\times$ 20, 15 $\times$ 20, 14 $\times$ 20, 14 $\times$ 19, 17 $\times$ 19, 10 $\times$ 17, 13 $\times$ 17, 5 $\times$ 17, 16 $\times$ 17, 20 $\times$ 16, 5 $\times$ 16, 3 $\times$ 15, 5 $\times$ 14, 18 $\times$ 14, 10 $\times$ 13, 14 $\times$ 12, 11 $\times$ 11, 2 $\times$ 10, 7 $\times$ 9, 14 $\times$ 8, 13 $\times$ 8, 9 $\times$ 7, 7 $\times$ 7, 16 $\times$ 6, 20 $\times$ 5, 2 $\times$ 5, 9 $\times$ 3, 17 $\times$ 3, 5 $\times$ 3, 4 $\times$ 2
2	2	2 (63 $\times$ 60)	23	22 $\times$ 23, 17 $\times$ 23, 13 $\times$ 22, 7 $\times$ 22, 9 $\times$ 21, 5 $\times$ 20, 20 $\times$ 20, 6 $\times$ 19, 7 $\times$ 17, 14 $\times$ 16, 12 $\times$ 14, 15 $\times$ 14, 20 $\times$ 14, 9 $\times$ 10, 16 $\times$ 9, 20 $\times$ 9, 18 $\times$ 8, 3 $\times$ 8, 12 $\times$ 7, 18 $\times$ 6, 20 $\times$ 4, 9 $\times$ 3, 6 $\times$ 3
3	7	24 $\times$ 14, 2 (18 $\times$ 10), 24 $\times$ 13, 3 (13 $\times$ 10)	17	3 $\times$ 5, 2 $\times$ 5, 4 $\times$ 3, 6 $\times$ 3, 3 $\times$ 3, 7 $\times$ 3, 2 $\times$ 2, 6 $\times$ 2, 9 $\times$ 2, 4 $\times$ 2, 1 $\times$ 2, 7 $\times$ 2, 4 $\times$ 1, 6 $\times$ 1, 2 $\times$ 1, 9 $\times$ 1, 7 $\times$ 1
4	11	2 (24 $\times$ 14), 3 (18 $\times$ 10), 2 (24 $\times$ 13), 4 (13 $\times$ 10)	27	3 $\times$ 5, 1 $\times$ 5, 7 $\times$ 5, 2 $\times$ 5, 3 $\times$ 4, 7 $\times$ 4, 4 $\times$ 4, 4 $\times$ 3, 1 $\times$ 3, 6 $\times$ 3, 8 $\times$ 3, 3 $\times$ 3, 7 $\times$ 3, 5 $\times$ 2, 6 $\times$ 2, 2 $\times$ 2, 9 $\times$ 2, 4 $\times$ 2, 1 $\times$ 2, 7 $\times$ 2, 8 $\times$ 1, 2 $\times$ 1, 6 $\times$ 1, 7 $\times$ 1, 9 $\times$ 1, 4 $\times$ 1, 5 $\times$ 1
5	16	2 (24 $\times$ 14), 5 (18 $\times$ 10), 3 (24 $\times$ 13), 6 (13 $\times$ 10)	37	3 $\times$ 5, 1 $\times$ 5, 7 $\times$ 5, 2 $\times$ 5, 8 $\times$ 5, 6 $\times$ 4, 5 $\times$ 4, 3 $\times$ 4, 7 $\times$ 4, 4 $\times$ 4, 1 $\times$ 4, 2 $\times$ 4, 4 $\times$ 3, 5 $\times$ 3, 1 $\times$ 3, 2 $\times$ 3, 6 $\times$ 3, 8 $\times$ 3, 3 $\times$ 3, 7 $\times$ 3, 6 $\times$ 2, 9 $\times$ 2, 7 $\times$ 2, 4 $\times$ 2, 5 $\times$ 2, 1 $\times$ 2, 2 $\times$ 2, 3 $\times$ 2, 6 $\times$ 1, 2 $\times$ 1, 4 $\times$ 1, 8 $\times$ 1, 5 $\times$ 1, 9 $\times$ 1, 3 $\times$ 1, 1 $\times$ 1, 7 $\times$ 1
6	8	2 (24 $\times$ 14), 4 (18 $\times$ 10), 2 (24 $\times$ 13)	34	3 $\times$ 5, 1 $\times$ 5, 7 $\times$ 5, 2 $\times$ 5, 8 $\times$ 5, 6 $\times$ 4, 3 $\times$ 4, 7 $\times$ 4, 4 $\times$ 4, 1 $\times$ 4, 2 $\times$ 4, 4 $\times$ 3, 5 $\times$ 3, 1 $\times$ 3, 6 $\times$ 3, 8 $\times$ 3, 3 $\times$ 3, 7 $\times$ 3, 6 $\times$ 2, 7 $\times$ 2, 9 $\times$ 2, 5 $\times$ 2, 4 $\times$ 2, 2 $\times$ 2, 1 $\times$ 2, 3 $\times$ 2, 9 $\times$ 1, 7 $\times$ 1, 8 $\times$ 1, 5 $\times$ 1, 2 $\times$ 1, 6 $\times$ 1, 4 $\times$ 1, 1 $\times$ 1
7	3	24 $\times$ 14, 18 $\times$ 10, 24 $\times$ 13	11	2 (2 $\times$ 2), 6 $\times$ 2, 4 $\times$ 2, 1 $\times$ 2, 2 (7 $\times$ 1), 4 $\times$ 1, 9 $\times$ 1, 6 $\times$ 1, 2 $\times$ 1
8	10	3 (28 $\times$ 17), 3 (16 $\times$ 27), 4 (13 $\times$ 23)	19	1 $\times$ 10, 6 $\times$ 10, 4 $\times$ 9, 8 $\times$ 9, 9 $\times$ 9, 5 $\times$ 8, 2 $\times$ 7, 4 $\times$ 6, 10 $\times$ 6, 6 $\times$ 6, 10 $\times$ 5, 5 $\times$ 5, 8 $\times$ 5, 4 $\times$ 4, 7 $\times$ 4, 10 $\times$ 4, 6 $\times$ 4, 7 $\times$ 3, 10 $\times$ 2
9	4	3 (19 $\times$ 10), 19 $\times$ 26	17	2 $\times$ 9, 2 $\times$ 8, 5 $\times$ 8, 3 $\times$ 8, 4 $\times$ 7, 5 $\times$ 7, 4 $\times$ 6, 6 $\times$ 4, 2 $\times$ 4, 3 $\times$ 4, 4 $\times$ 3, 2 $\times$ 3, 6 $\times$ 2, 5 $\times$ 2, 1 $\times$ 1, 2 $\times$ 1, 3 $\times$ 1
10	6	2 (290 $\times$ 106), 2 (148 $\times$ 183), 2 (194 $\times$ 132)	13	2 (63 $\times$ 59), 63 $\times$ 55, 48 $\times$ 48, 17 $\times$ 43, 98 $\times$ 40, 38 $\times$ 35, 2(114 $\times$ 33), 24 $\times$ 23, 62 $\times$ 19, 2(110 $\times$ 11)
11	13	5 (25 $\times$ 21), 5 (27 $\times$ 19), 3 (30 $\times$ 24)	37	8 (11 $\times$ 7), 5 (7 $\times$ 5), 9 (9 $\times$ 5), 5 (10 $\times$ 5), 10 (3 $\times$ 2)
12	1	14 $\times$ 19	12	12 (2 $\times$ 4)
13	7	2 (21 $\times$ 24), 5 (10 $\times$ 18)	32	15 (4 $\times$ 7), 17 (1 $\times$ 4)
14	4	24 $\times$ 14, 2 (18 $\times$ 10), 24 $\times$ 13	28	12 (7 $\times$ 1), 7 (6 $\times$ 1), 9 (4 $\times$ 1)

Table 18 (continued)

Inst.	Objects		Items	
	S	Width × Height	I'	Width × Height
15	12	4 (26 × 19), 4 (22 × 23), 4 (30 × 17)	34	13 (7 × 6), 10 (9 × 4), 11 (11 × 3)
16	19	6 (30 × 11), 6 (27 × 13), 7 (12 × 26)	21	10 (8 × 10), 9 (10 × 3), 2 (11 × 2)
17	1	14 × 19	17	7 (2 × 4), 10 (1 × 3)
18	5	3 (22 × 17), 2 (14 × 30)	24	14 (2 × 11), 10 (5 × 5)
19	16	4 (30 × 22), 4 (30 × 24), 8 (10 × 21)	41	15 (9 × 7), 11 (11 × 6), 15 (1 × 5)
20	9	5 (22 × 17), 4 (14 × 30)	21	8 (2 × 11), 7 (8 × 9), 6 (5 × 5)

Funding This research was funded by the São Paulo Research Foundation (*Fundação de Amparo à Pesquisa do Estado de São Paulo*) FAPESP (Grant Numbers 2019/25041-8, 2018/16600-0, 2018/07240-0 and 2016/01860-1) and the National Council for Scientific and Technological Development (*Conselho Nacional de Desenvolvimento Científico e Tecnológico*) CNPq (Grant Numbers 317460/2021-8, 421130/2018-0 and 306558/2018-1). This work is partially financed by the ERDF—European Regional Development Fund through the Operational Programme for Competitiveness and Internationalisation—COMPETE 2020 Programme and by National Funds through the Portuguese funding agency, FCT—Fundação para a Ciência e a Tecnologia, I.P., within project POCI-01-0145-FEDER-029609.

Availability of data and material The full description of all instances used in the computational tests can be found at http://data.mendeley.com/datasets/ddc79swng4/1.

Code availability Not applicable.

Declarations

Conflict of interest Not applicable.

References

Andrade R, Birgin EG, Morabito R, Ronconi DP (2014) MIP models for two-dimensional non-guillotine cutting problems with usable leftovers. J Oper Res Soc 65(11):1649–1663. https://doi.org/10.1057/jors.2013.108

Andrade R, Birgin EG, Morabito R (2016) Two-stage two-dimensional guillotine cutting stock problems with usable leftover. Int Trans Oper Res 23:121–145. https://doi.org/10.1111/itor.12077

Arenales MN, Cherri AC, do Nascimento DN, Vianna ACG (2015) A new mathematical model for the cutting stock/leftover problem. Pesqui Oper 35(3):509–522. https://doi.org/10.1590/0101-7438.2015.035.03.0509

Birgin EG, Romão OC, Ronconi DP (2020) The multiperiod two-dimensional non-guillotine cutting stock problem with usable leftovers. Int Trans Oper Res 27(3):1392–1418. https://doi.org/10.1111/itor.12648

Bouaine A, Lebbar M, Ha MA (2018) Minimization of the wood wastes for an industry of furnishing: a two dimensional cutting stock problem. Manag Prod Eng Rev 9(2):42–51. https://doi.org/10.24425/119524

Cherri AC (2009) Algumas extensões do problema de corte de estoque com sobras de material aproveitáveis. Doctoral thesis, ICMC-USP, São Carlos

Cherri AC, Arenales MN, Yanasse HH, Poldi KC, Vianna ACG (2014) The one-dimensional cutting stock problem with usable leftovers—a survey. Eur J Oper Res 236(2):395–402

Christofides N, Hadjiconstantinou E (1995) An exact algorithm for orthogonal 2-D cutting problems using guillotine cuts. Eur J Oper Res 83(1):21–38. https://doi.org/10.1016/0377-2217(93)E0277-5

Cintra GF, Miyazawa FK, Wakabayashi Y, Xavier EC (2008) Algorithms for two-dimensional cutting stock and strip packing problems using dynamic programming and column generation. Eur J Oper Res 191:61–85. https://doi.org/10.1016/j.ejor.2007.08.007

Clautiaux F, Sadykov R, Vanderbeck F, Viaud Q (2019) Pattern-based diving heuristics for a two-dimensional guillotine cutting-stock problem with leftovers. EURO J Comput Optim 7:265–297. https://doi.org/10.1007/s13675-019-00113-9

Coelho KR, Cherri AC, Baptista EC, Jabbour CJC, Soler EM (2017) Sustainable operations: The cutting stock problem with usable leftovers from a sustainable perspective. J Clean Prod 167:545–552. https://doi.org/10.1016/j.jclepro.2017.08.153

do Nascimento DN, de Araújo SA, Cherri AC (2020) Integrated lot-sizing and one-dimensional cutting stock problem with usable leftovers. Ann Oper Res. https://doi.org/10.1007/s10479-020-03772-9

Dusberger F, Raidl GR (2015) Solving the 3-staged 2-dimensional cutting stock problem by dynamic programming and variable neighborhood search. Electron Notes Discrete Math 47:133–140. https://doi.org/10.1016/j.endm.2014.11.018

Furini F, Malaguti E (2013) Models for the two-dimensional two-stage cutting stock problem with multiple stock size. Comput Oper Res 40(8):1953–1962. https://doi.org/10.1016/j.cor.2013.02.026

Furini F, Malaguti E, Durán RM, Persiani A, Toth P (2012) A column generation heuristic for the two-dimensional two-staged guillotine cutting stock problem with multiple stock size. Eur J Oper Res 218(1):251–260. https://doi.org/10.1016/j.ejor.2011.10.018

Furini F, Malaguti E, Thomopulos D (2016) Modeling two-dimensional guillotine cutting problems via integer programming. INFORMS J Comput 28:736–751. https://doi.org/10.1287/ijoc.2016.0710

Gilmore PC, Gomory RE (1961) A linear programming approach to the cutting-stock problem. Oper Res 9(6):849–859. https://doi.org/10.1287/opre.9.6.849

Gilmore PC, Gomory RE (1963) A linear programming approach to the cutting-stock problem—part II. Oper Res 11(6):863–888. https://doi.org/10.1287/opre.11.6.863

Gilmore PC, Gomory RE (1965) Multistage cutting stock problems of two and more dimensions. Oper Res 13(1):94–120. https://doi.org/10.1287/opre.13.1.94

Kwon SJ, Joung S, Lee K (2019) Comparative analysis of pattern-based models for the two-dimensional two-stage guillotine cutting stock problem. Comput Oper Res 109:159–169. https://doi.org/10.1016/j.cor.2019.05.005

Li F, Chen Y, Hu X (2022) Manufacturing-oriented silicon steel coil lengthwise cutting stock problem with useable leftover. Eng Comput (Swansea) 39(2):477–492. https://doi.org/10.1108/EC-11-2020-0660

Liu L, Liu X, Pei J, Fan W, Pardalos PM (2017) A study on decision making of cutting stock with frustum of cone bars. Oper Res Int J 17:187–204. https://doi.org/10.1007/s12351-015-0221-x

Lodi A, Monaci M (2003) Integer linear programming models for 2-staged two-dimensional Knapsack problems. Math Program 94(2):257–278. https://doi.org/10.1007/s10107-002-0319-9

Martin M, Hokama PH, Morabito R, Munari P (2020) The constrained two-dimensional guillotine cutting problem with defects: an ILP formulation, a Benders decomposition and a CP-based algorithm. Int J Prod Res 58(9):2712–2729. https://doi.org/10.1080/00207543.2019.1630773

Morabito R (1989) Corte de estoque bidimensional. Dissertation, ICMC–USP, São Carlos

Silva E, Alvelos F, de Carvalho JV (2010) An integer programming model for two- and three-stage two-dimensional cutting stock problems. Eur J Oper Res 205(3):699–708. https://doi.org/10.1016/j.ejor.2010.01.039

Suliman S (2006) A sequential heuristic procedure for the two-dimensional cutting-stock problem. Int J Prod Econ 99(1):177–185. https://doi.org/10.1016/j.ijpe.2004.12.017

Sumetthapiwat S, Intiyot B, Jeenanunta C (2020) A column generation on two-dimensional cutting stock problem with fixed-size usable leftover and multiple stock sizes. Int J Logist Syst Manag 35(2):273–288. https://doi.org/10.1504/IJLSM.2020.104781

Tomat L, Gradisar M (2017) One-dimensional stock cutting: optimization of usable leftovers in consecutive orders. Cent Eur J Oper Res 25(2):473–489. https://doi.org/10.1007/s10100-017-0466-y

Wang PY (1983) Two algorithms for constrained two-dimensional cutting stock problems. INFORMS Oper Res 31(3):573—586. https://www.jstor.org/stable/170624

Wang D, Xiao F, Zhou L, Liang Z (2020) Two-dimensional skiving and cutting stock problem with setup cost based on column-and-row generation. Eur J Oper Res 286(2):547–563. https://doi.org/10.1016/j.ejor.2020.03.060

Wuttke DA, Heese HS (2018) Two-dimensional cutting stock problem with sequence dependent setup times. Eur J Oper Res 265(1):303–315. https://doi.org/10.1016/j.ejor.2017.07.036